data.world

Learn Less, Parse More

Using Grammars to Post-Process Generated Text

Marie Chatfield Rivas (*they/them*)
Staff Engineer

Large language models will **never be fully reliable.**

Parse text into **well-defined values**,

post-process to **increase correctness.**

Parse **standard grammars**

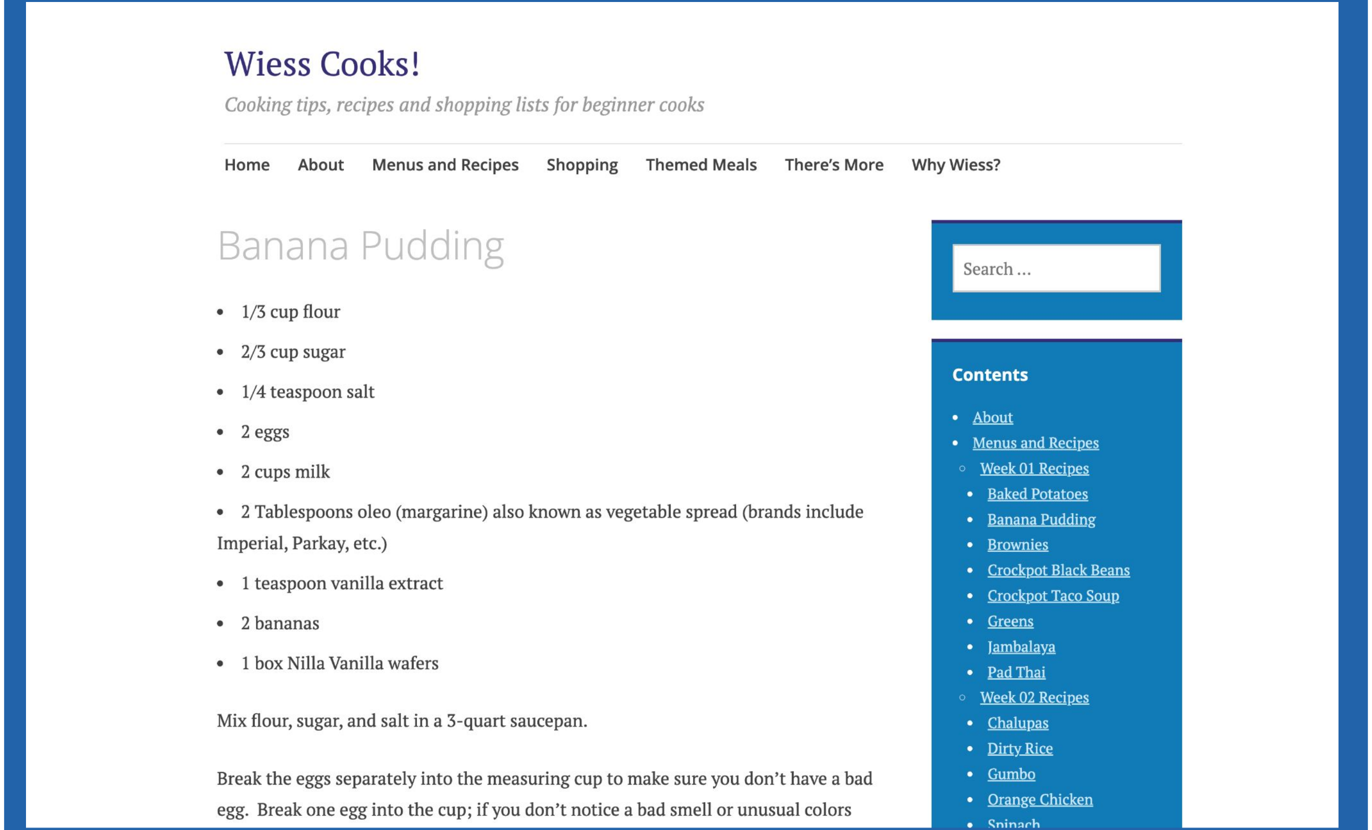
Post-process for **reliability**

Craft **domain-specific** grammars

Optimize grammars for **LLM cooperation**

Support **multiple languages** with post-processing

wiesscooks.blogs.rice.edu



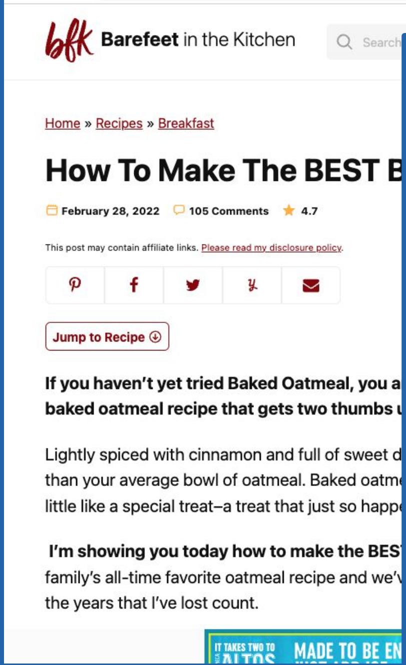
Banana Pudding

- 1/3 cup flour
- 2/3 cup sugar
- 1/4 teaspoon salt
- 2 eggs
- 2 cups milk
- 2 Tablespoons oleo (margarine) also known as vegetable spread (brands include Imperial, Parkay, etc.)
- 1 teaspoon vanilla extract
- 2 bananas
- 1 box Nilla Vanilla wafers

Mix flour, sugar, and salt in a 3-quart saucepan.

Break the eggs separately into the measuring cup to make sure you don't have a bad egg. Break one egg into the cup; if you don't notice a bad smell or unusual colors

barefeetinthekitchen.com



Home » Recipes » Breakfast

How To Make The BEST B

February 28, 2022 105 Comments ★ 4.7

This post may contain affiliate links. Please read my disclosure policy.

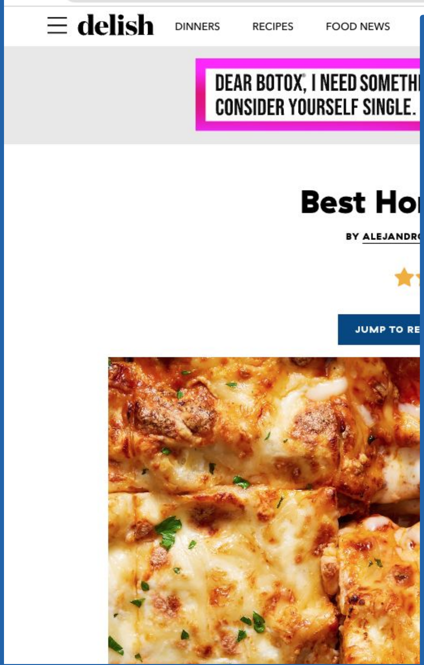
Jump to Recipe

If you haven't yet tried Baked Oatmeal, you a
baked oatmeal recipe that gets two thumbs u

Lightly spiced with cinnamon and full of sweet d
than your average bowl of oatmeal. Baked oatm
little like a special treat–a treat that just so happ

I'm showing you today how to make the BES
family's all-time favorite oatmeal recipe and we'
the years that I've lost count.

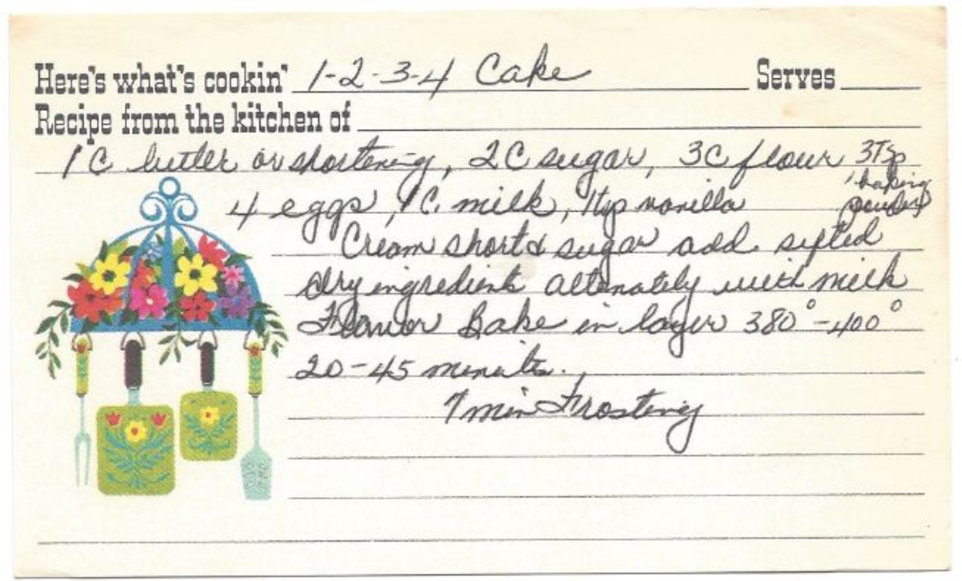
delish.com



Best Ho

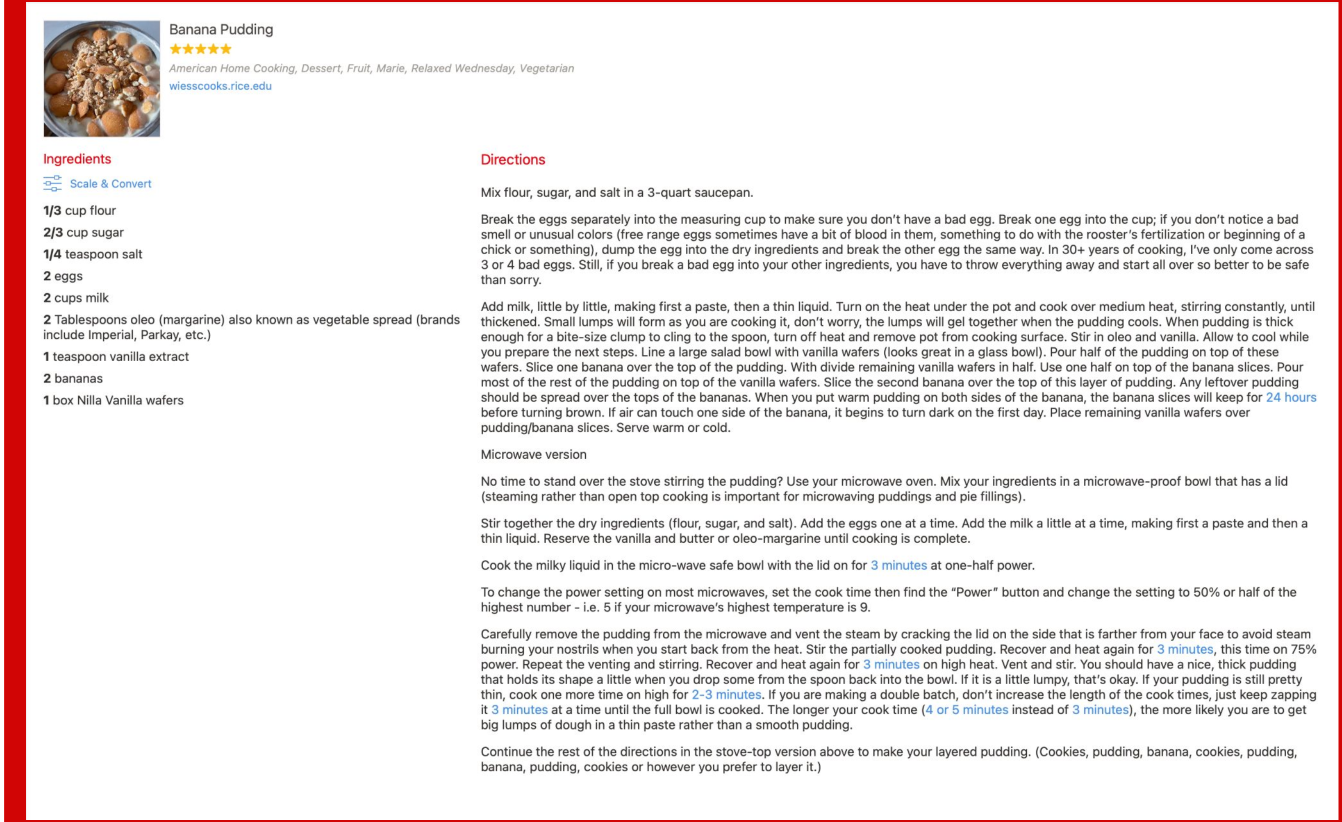
BY ALEJANDR

Grandma's Recipes



Here's what's cookin' 1-2-3-4 Cake Serves
Recipe from the kitchen of
1C butter or shortening, 2C sugar, 3C flour 3Tsp baking powder
4 eggs, 1C milk, 1tsp vanilla
Cream short & sugar add sifted
dry ingredients alternately with milk
Flavor Bake in layers 380°-400°
20-45 minutes.
7 min frosting

Paprika Recipe Manager



Banana Pudding

★★★★★

American Home Cooking, Dessert, Fruit, Marie, Relaxed Wednesday, Vegetarian

wiesscooks.rice.edu

Ingredients

Scale & Convert

1/3 cup flour

2/3 cup sugar

1/4 teaspoon salt

2 eggs

2 cups milk

2 Tablespoons oleo (margarine) also known as vegetable spread (brands include Imperial, Parkay, etc.)

1 teaspoon vanilla extract

2 bananas

1 box Nilla Vanilla wafers

Directions

Mix flour, sugar, and salt in a 3-quart saucepan.

Break the eggs separately into the measuring cup to make sure you don't have a bad egg. Break one egg into the cup; if you don't notice a bad smell or unusual colors (free range eggs sometimes have a bit of blood in them, something to do with the rooster's fertilization or beginning of a chick or something), dump the egg into the dry ingredients and break the other egg the same way. In 30+ years of cooking, I've only come across 3 or 4 bad eggs. Still, if you break a bad egg into your other ingredients, you have to throw everything away and start all over so better to be safe than sorry.

Add milk, little by little, making first a paste, then a thin liquid. Turn on the heat under the pot and cook over medium heat, stirring constantly, until thickened. Small lumps will form as you are cooking it, don't worry, the lumps will gel together when the pudding cools. When pudding is thick enough for a bite-size clump to cling to the spoon, turn off heat and remove pot from cooking surface. Stir in oleo and vanilla. Allow to cool while you prepare the next steps. Line a large salad bowl with vanilla wafers (looks great in a glass bowl). Pour half of the pudding on top of these wafers. Slice one banana over the top of the pudding. With divide remaining vanilla wafers in half. Use one half on top of the banana slices. Pour most of the rest of the pudding on top of the vanilla wafers. Slice the second banana over the top of this layer of pudding. Any leftover pudding should be spread over the tops of the bananas. When you put warm pudding on both sides of the banana, the banana slices will keep for 24 hours before turning brown. If air can touch one side of the banana, it begins to turn dark on the first day. Place remaining vanilla wafers over pudding/banana slices. Serve warm or cold.

Microwave version

No time to stand over the stove stirring the pudding? Use your microwave oven. Mix your ingredients in a microwave-proof bowl that has a lid (steaming rather than open top cooking is important for microwaving puddings and pie fillings).

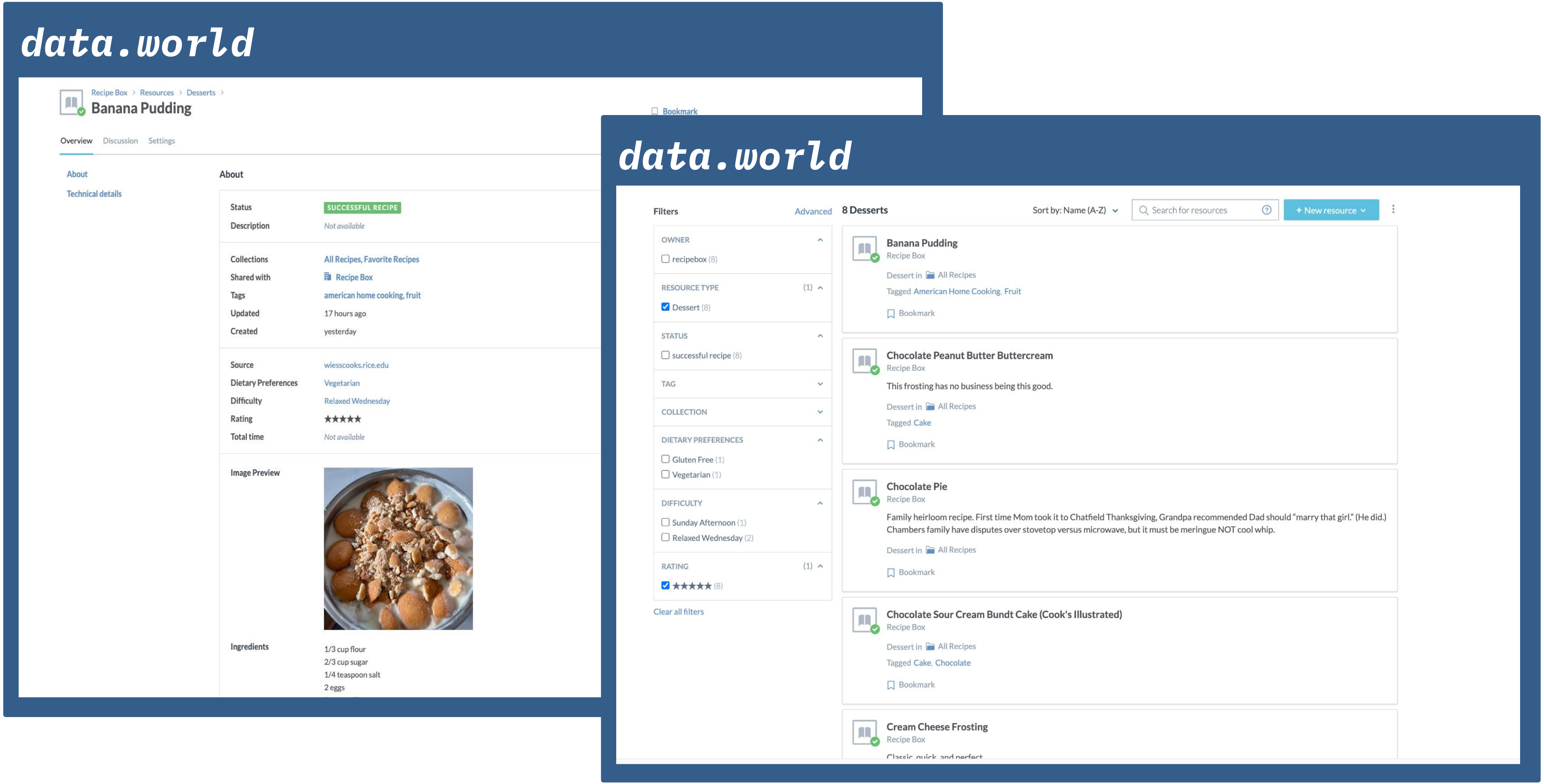
Stir together the dry ingredients (flour, sugar, and salt). Add the eggs one at a time. Add the milk a little at a time, making first a paste and then a thin liquid. Reserve the vanilla and butter or oleo-margarine until cooking is complete.

Cook the milky liquid in the micro-wave safe bowl with the lid on for 3 minutes at one-half power.

To change the power setting on most microwaves, set the cook time then find the "Power" button and change the setting to 50% or half of the highest number - i.e. 5 if your microwave's highest temperature is 9.

Carefully remove the pudding from the microwave and vent the steam by cracking the lid on the side that is farther from your face to avoid steam burning your nostrils when you start back from the heat. Stir the partially cooked pudding. Recover and heat again for 3 minutes, this time on 75% power. Repeat the venting and stirring. Recover and heat again for 3 minutes on high heat. Vent and stir. You should have a nice, thick pudding that holds its shape a little when you drop some from the spoon back into the bowl. If it is a little lumpy, that's okay. If your pudding is still pretty thin, cook one more time on high for 2-3 minutes. If you are making a double batch, don't increase the length of the cook times, just keep zapping it 3 minutes at a time until the full bowl is cooked. The longer your cook time (4 or 5 minutes instead of 3 minutes), the more likely you are to get big lumps of dough in a thin paste rather than a smooth pudding.

Continue the rest of the directions in the stove-top version above to make your layered pudding. (Cookies, pudding, banana, cookies, pudding, banana, pudding, cookies or however you prefer to layer it.)



data.world

Recipe Box › Resources › Desserts ›

Banana Pudding

Bookmark

Overview Discussion Settings

About
Technical details

About

Status	SUCCESSFUL RECIPE
Description	*Not available*
Collections	All Recipes, Favorite Recipes
Shared with	Recipe Box
Tags	american home cooking, fruit
Updated	17 hours ago
Created	yesterday
Source	wiesscooks.rice.edu
Dietary Preferences	Vegetarian
Difficulty	Relaxed Wednesday
Rating	★★★★★
Total time	*Not available*
Image Preview	
Ingredients	1/3 cup flour 2/3 cup sugar 1/4 teaspoon salt 2 eggs

data.world

Filters Advanced

OWNER
☐ recipebox (8)

RESOURCE TYPE (1)
☑ Dessert (8)

STATUS
☐ successful recipe (8)

TAG

COLLECTION

DIETARY PREFERENCES
☐ Gluten Free (1)
☐ Vegetarian (1)

DIFFICULTY
☐ Sunday Afternoon (1)
☐ Relaxed Wednesday (2)

RATING (1)
☑ ★★★★★ (8)

Clear all filters

8 Desserts Sort by: Name (A-Z) Search for resources + New resource

Banana Pudding
Recipe Box
Dessert in All Recipes
Tagged American Home Cooking, Fruit
Bookmark

Chocolate Peanut Butter Buttercream
Recipe Box
This frosting has no business being this good.
Dessert in All Recipes
Tagged Cake
Bookmark

Chocolate Pie
Recipe Box
Family heirloom recipe. First time Mom took it to Chatfield Thanksgiving, Grandpa recommended Dad should "marry that girl." (He did.) Chambers family have disputes over stovetop versus microwave, but it must be meringue NOT cool whip.
Dessert in All Recipes
Bookmark

Chocolate Sour Cream Bundt Cake (Cook's Illustrated)
Recipe Box
Dessert in All Recipes
Tagged Cake, Chocolate
Bookmark

Cream Cheese Frosting
Recipe Box
Classic, quick, and perfect

Recipes

Banana Pudding

12 servings
Total time: 1hr

Chocolate Peanut Butter Buttercream

Chocolate Pie

10 servings
Total time: 1.5hr

Ingredients

All-purpose flour
Salt
Brown sugar
Granulated sugar
Milk (whole)
Evaporated milk
Eggs
Egg yolks
Vanilla extract
Bananas
Margarine
Nilla wafers

Recipes

Banana Pudding

12 servings
Total time: 1hr

Chocolate Peanut Butter Buttercream

Chocolate Pie

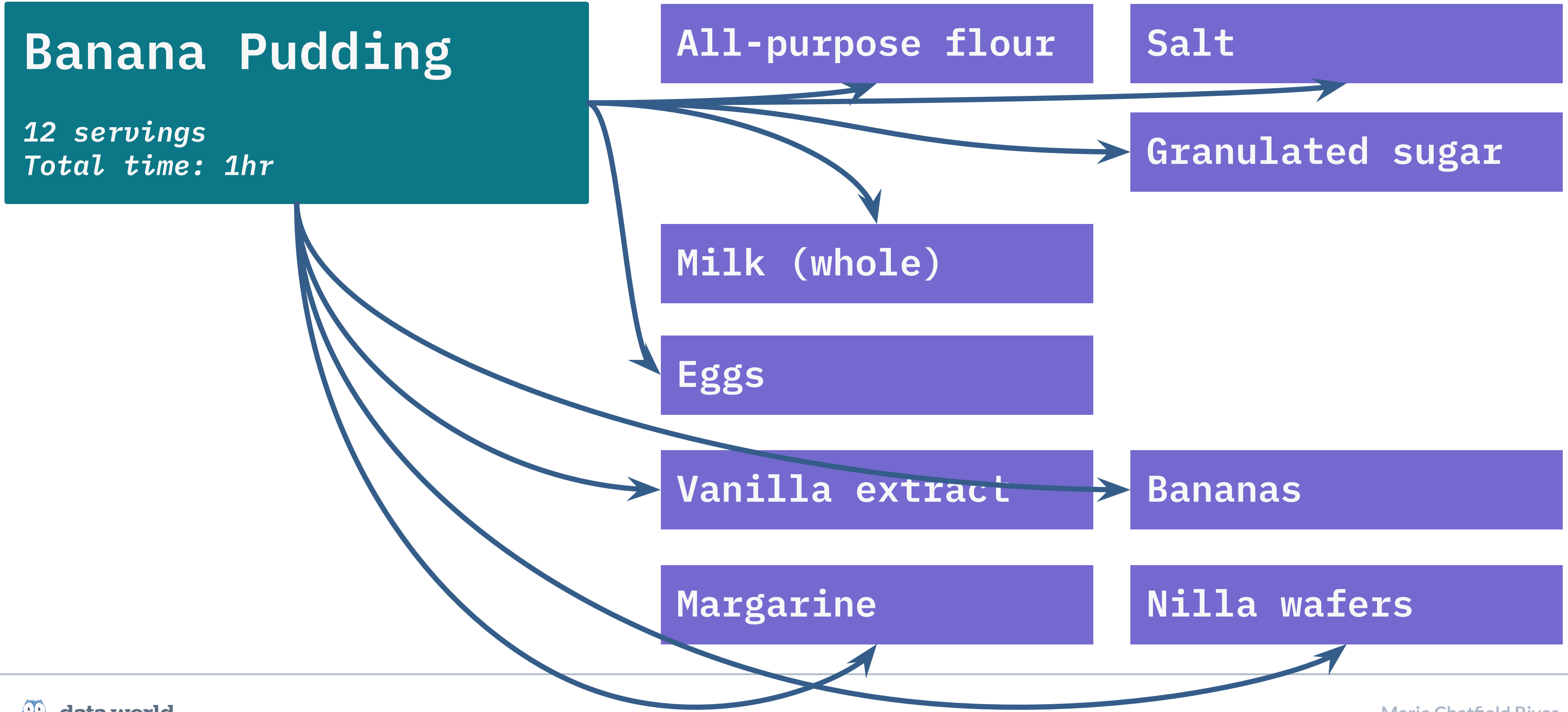
10 servings
Total time: 1.5hr

Ingredients

- All-purpose flour
- Brown sugar
- Milk (whole)
- Eggs
- Vanilla extract
- Margarine
- Salt
- Granulated sugar
- Evaporated milk
- Egg yolks
- Bananas
- Nilla wafers

Recipes

Ingredients



Banana Pudding

12 servings
Total time: 1hr

All-purpose flour

Salt

Granulated sugar

Milk (whole)

Eggs

Vanilla extract

Bananas

Margarine

Nilla wafers

Recipes

Banana Pudding

12 servings
Total time: 1hr

1/3 cup flour
2/3 cup sugar
1/4 teaspoon salt
2 eggs
2 cups milk
2 Tablespoons oleo (margarine) also known as vegetable spread (brands include Imperial, Parkay, etc.)
1 teaspoon vanilla extract
2 bananas
1 box Nilla Vanilla wafers

Ingredients

- All-purpose flour
- Salt
- Granulated sugar
- Milk (whole)
- Eggs
- Vanilla extract
- Bananas
- Margarine
- Nilla wafers

Parse standard grammars

Large Language Model

This is a recipe. What are the ingredients as a list?

Banana Pudding

1/3 cup flour
2/3 cup sugar
1/4 teaspoon salt
2 eggs
…

Sure! Here are the ingredients for Banana Pudding:

- Flour
- Sugar
- Salt
- Eggs
- Milk
- Margarine (or vegetable spread)
- Vanilla extract
- Bananas
- Nilla Vanilla wafers (vanilla-flavored cookies)

Large Language Model

This is a recipe. Return standardized ingredients as
valid JSON in the format: `{ "ingredients": [] }`

```
Banana Pudding

1/3 cup flour
2/3 cup sugar
1/4 teaspoon salt
2 eggs
…
```

```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk (whole)",
    "Margarine",
    "Vanilla extract",
    "Bananas",
    "Nilla wafers"
  ]
}
```

JSON.parse(

```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk (whole)",
    "Margarine",
    "Vanilla extract",
    "Bananas",
    "Nilla wafers"
  ]
}
```

)

All-purpose flour

Salt

Granulated sugar

Milk (whole)

Eggs

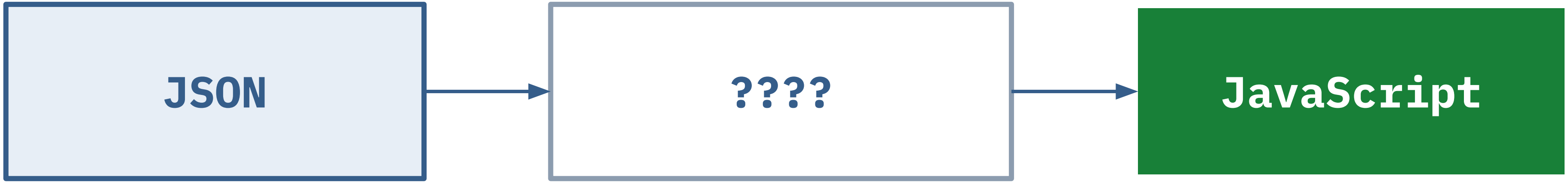
Vanilla extract

Bananas

Margarine

Nilla wafers

JSON.parse


JSON

????

JavaScript

JSON Grammar

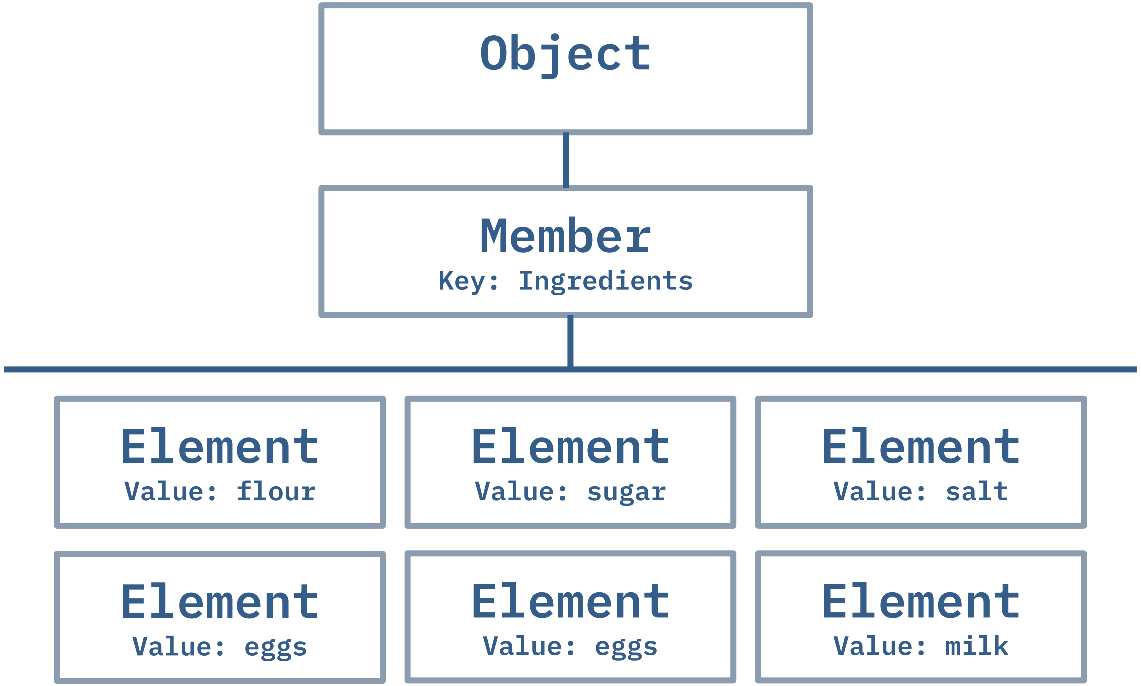
```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk",
    "Margarine",
    "Vanilla extract",
    "Bananas",
    "Nilla wafers"
  ]
}
```

```
<json>    ::= <element>
<element> ::= <space>? <value> <space>?

<value> ::=
     <object>
     <array>
     <string>
     <number>
     "true"
     "false"
     "null"
```

Abstract Syntax Tree (AST)

```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk",
    "Margarine",
    "Vanilla extract",
    "Bananas",
    "Nilla wafers"
  ]
}
```



Object

Member
Key: Ingredients

Element
Value: flour

Element
Value: sugar

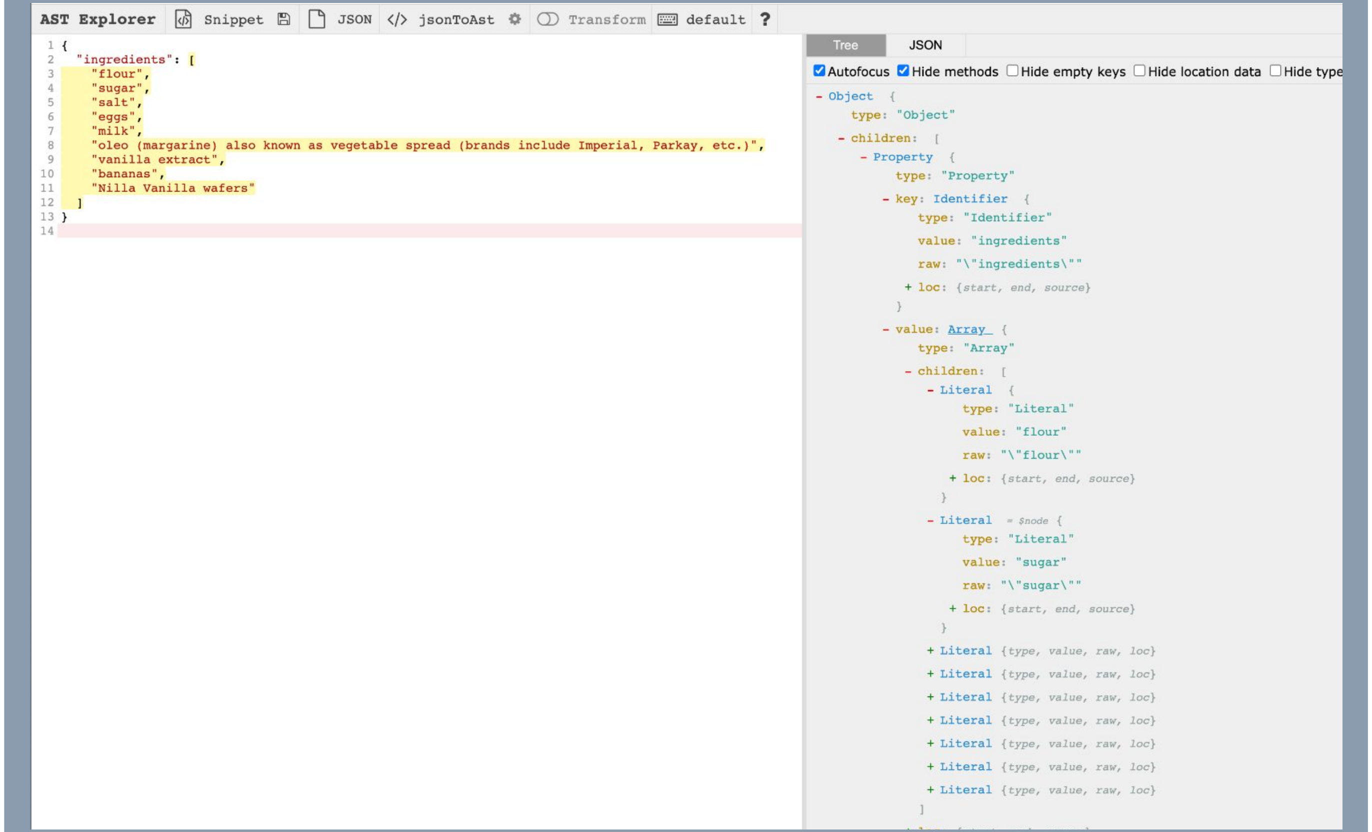
Element
Value: salt

Element
Value: eggs

Element
Value: eggs

Element
Value: milk

astexplorer.net

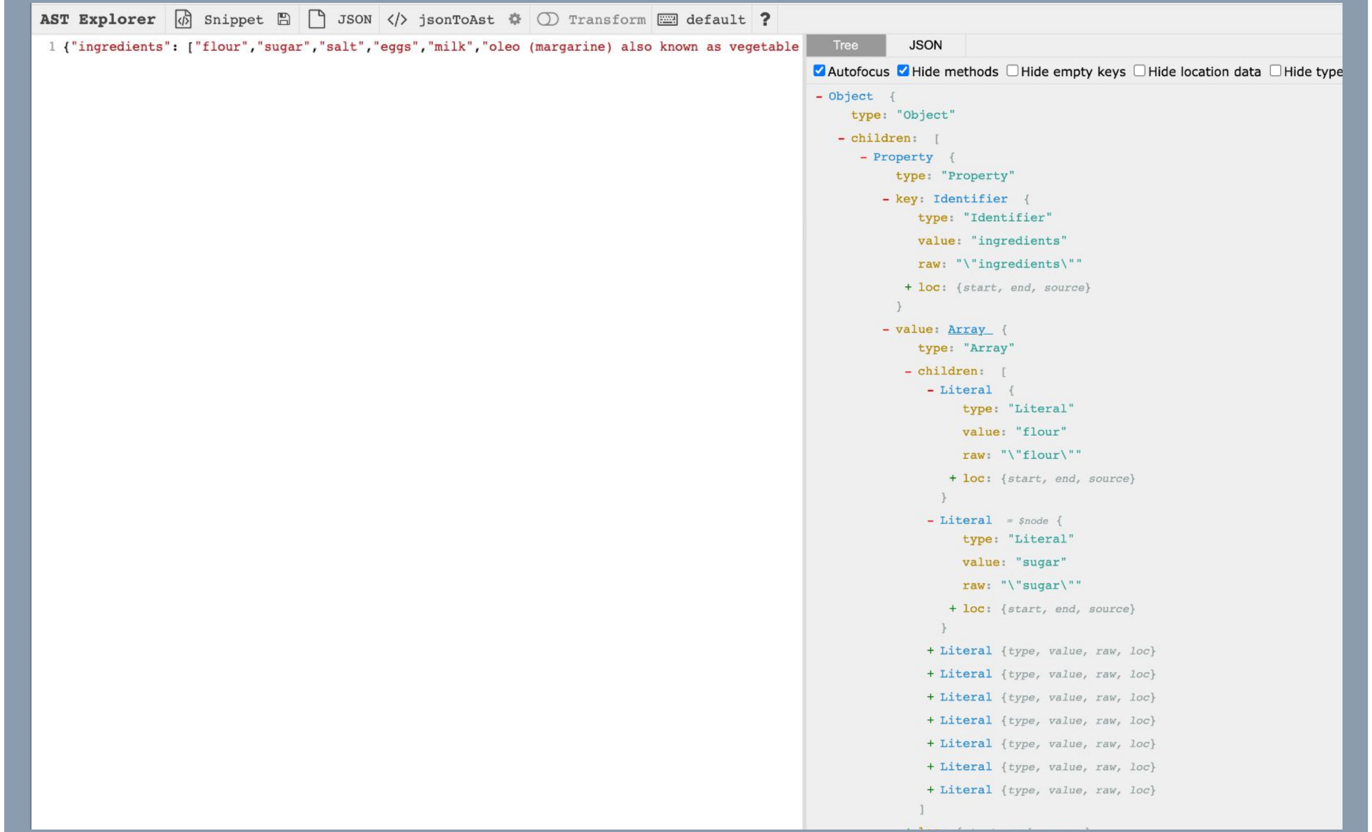


```
{
  "ingredients": [
    "flour",
    "sugar",
    "salt",
    "eggs",
    "milk",
    "oleo (margarine) also known as vegetable spread (brands include Imperial, Parkay, etc.)",
    "vanilla extract",
    "bananas",
    "Nilla Vanilla wafers"
  ]
}
```

☑ Autofocus ☑ Hide methods ☐ Hide empty keys ☐ Hide location data ☐ Hide type

```
- Object  {
    type: "Object"
  - children:  [
    - Property  {
        type: "Property"
      - key: Identifier  {
          type: "Identifier"
          value: "ingredients"
          raw: "\"ingredients\""
        + loc: {start, end, source}
        }
      - value: Array  {
          type: "Array"
        - children:  [
          - Literal  {
              type: "Literal"
              value: "flour"
              raw: "\"flour\""
            + loc: {start, end, source}
            }
          - Literal  = $node {
              type: "Literal"
              value: "sugar"
              raw: "\"sugar\""
            + loc: {start, end, source}
            }
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          ]
```

astexplorer.net

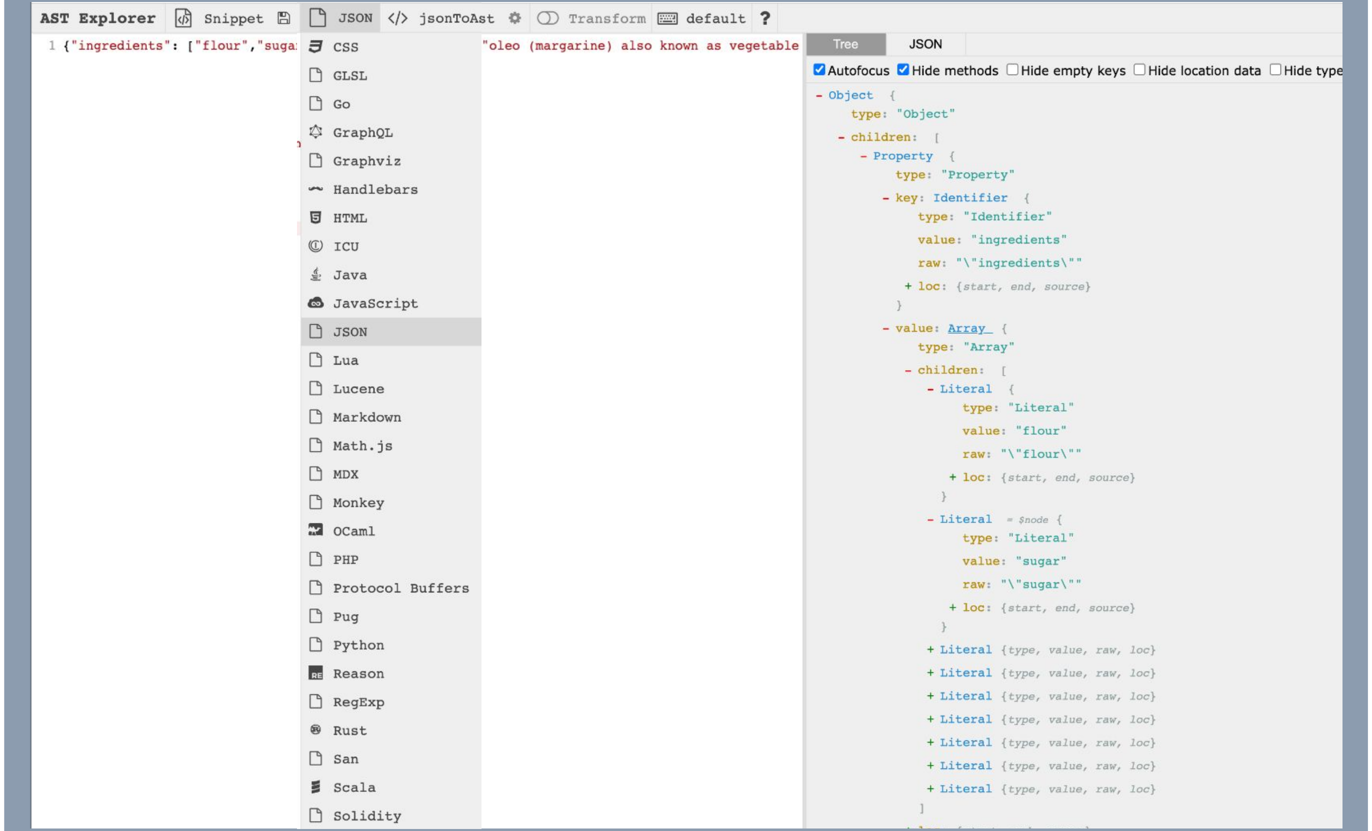


```
{"ingredients": ["flour","sugar","salt","eggs","milk","oleo (margarine) also known as vegetable
```

☑ Autofocus ☑ Hide methods ☐ Hide empty keys ☐ Hide location data ☐ Hide type

```
- Object  {
    type: "Object"
  - children:  [
    - Property  {
        type: "Property"
      - key: Identifier  {
          type: "Identifier"
          value: "ingredients"
          raw: "\"ingredients\""
        + loc: {start, end, source}
        }
      - value: Array  {
          type: "Array"
        - children:  [
          - Literal  {
              type: "Literal"
              value: "flour"
              raw: "\"flour\""
            + loc: {start, end, source}
            }
          - Literal  = $node {
              type: "Literal"
              value: "sugar"
              raw: "\"sugar\""
            + loc: {start, end, source}
            }
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          + Literal {type, value, raw, loc}
          ]
```

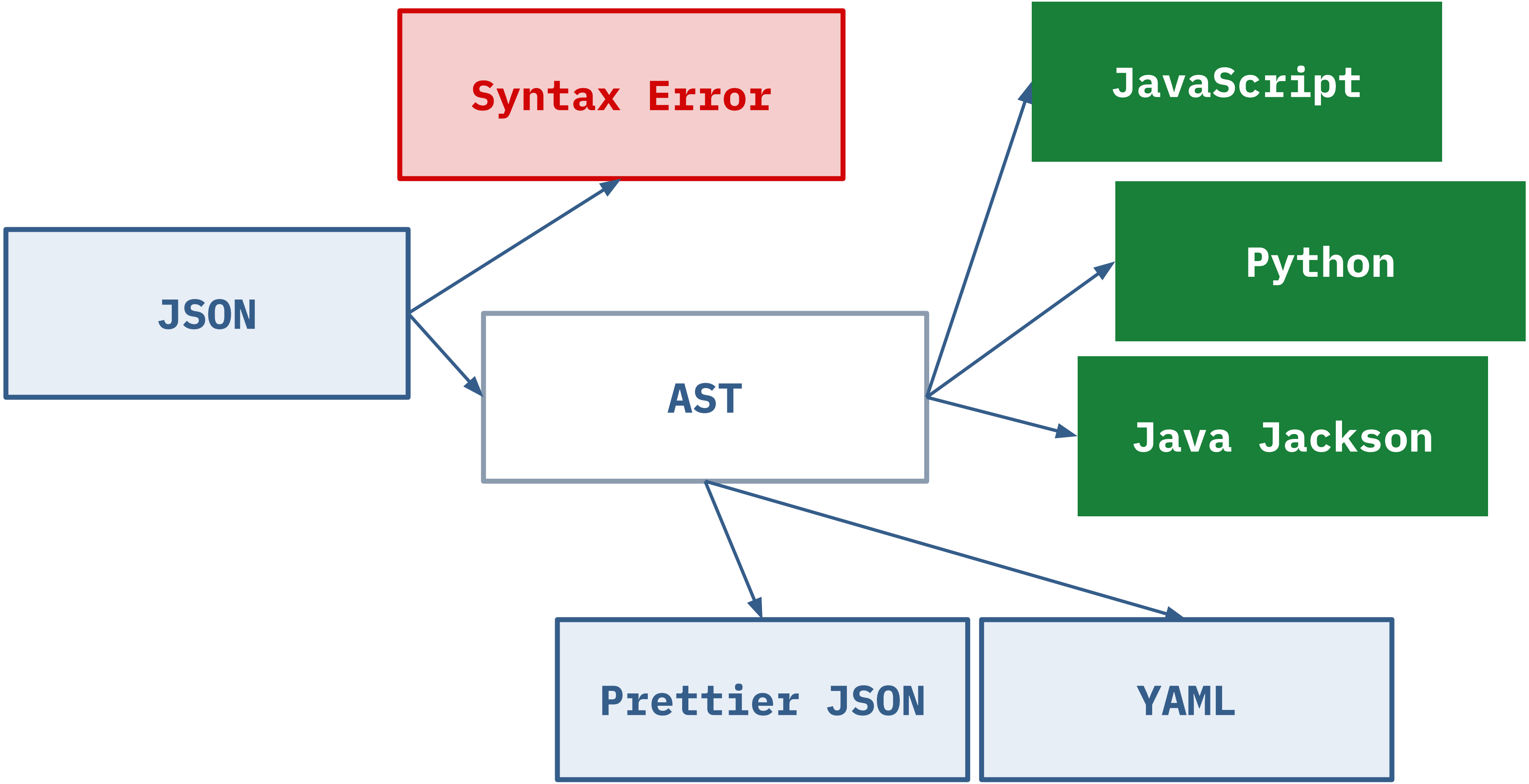
astexplorer.net



```
{"ingredients": ["flour","suga    "oleo (margarine) also known as vegetable
```

☑ Autofocus ☑ Hide methods ☐ Hide empty keys ☐ Hide location data ☐ Hide type

```
- Object  {
    type: "Object"
  - children:  [
     - Property  {
          type: "Property"
        - key: Identifier  {
             type: "Identifier"
             value: "ingredients"
             raw: "\"ingredients\""
           + loc: {start, end, source}
          }
        - value: Array  {
             type: "Array"
           - children:  [
              - Literal  {
                   type: "Literal"
                   value: "flour"
                   raw: "\"flour\""
                 + loc: {start, end, source}
                }
              - Literal  = $node {
                   type: "Literal"
                   value: "sugar"
                   raw: "\"sugar\""
                 + loc: {start, end, source}
                }
              + Literal {type, value, raw, loc}
              + Literal {type, value, raw, loc}
              + Literal {type, value, raw, loc}
              + Literal {type, value, raw, loc}
              + Literal {type, value, raw, loc}
              + Literal {type, value, raw, loc}
              + Literal {type, value, raw, loc}
             ]
```


Syntax Error
JSON
AST
JavaScript
Python
Java Jackson
Prettier JSON
YAML

Generate ingredients as JSON

Banana Pudding

12 servings
Total time: 1hr

1/3 cup flour
2/3 cup sugar
1/4 teaspoon salt
2 eggs
2 cups milk
2 Tablespoons oleo (margarine) also known as vegetable spread (brands include Imperial, Parkay, etc.)
1 teaspoon vanilla extract
2 bananas
1 box Nilla Vanilla wafers

Large Language Model

```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk (whole)",
    "Margarine (oleo)",
    "Vanilla extract",
    "Bananas",
    "Nilla vanilla wafers"
  ]
}
```

Parse JSON

Banana Pudding

12 servings
Total time: 1hr

1/3 cup flour
2/3 cup sugar
1/4 teaspoon salt
2 eggs
2 cups milk
2 Tablespoons oleo (margarine) also known as vegetable spread (brands include Imperial, Parkay, etc.)
1 teaspoon vanilla extract
2 bananas
1 box Nilla Vanilla wafers

All-purpose flour

Salt

Granulated sugar

Milk (whole)

Eggs

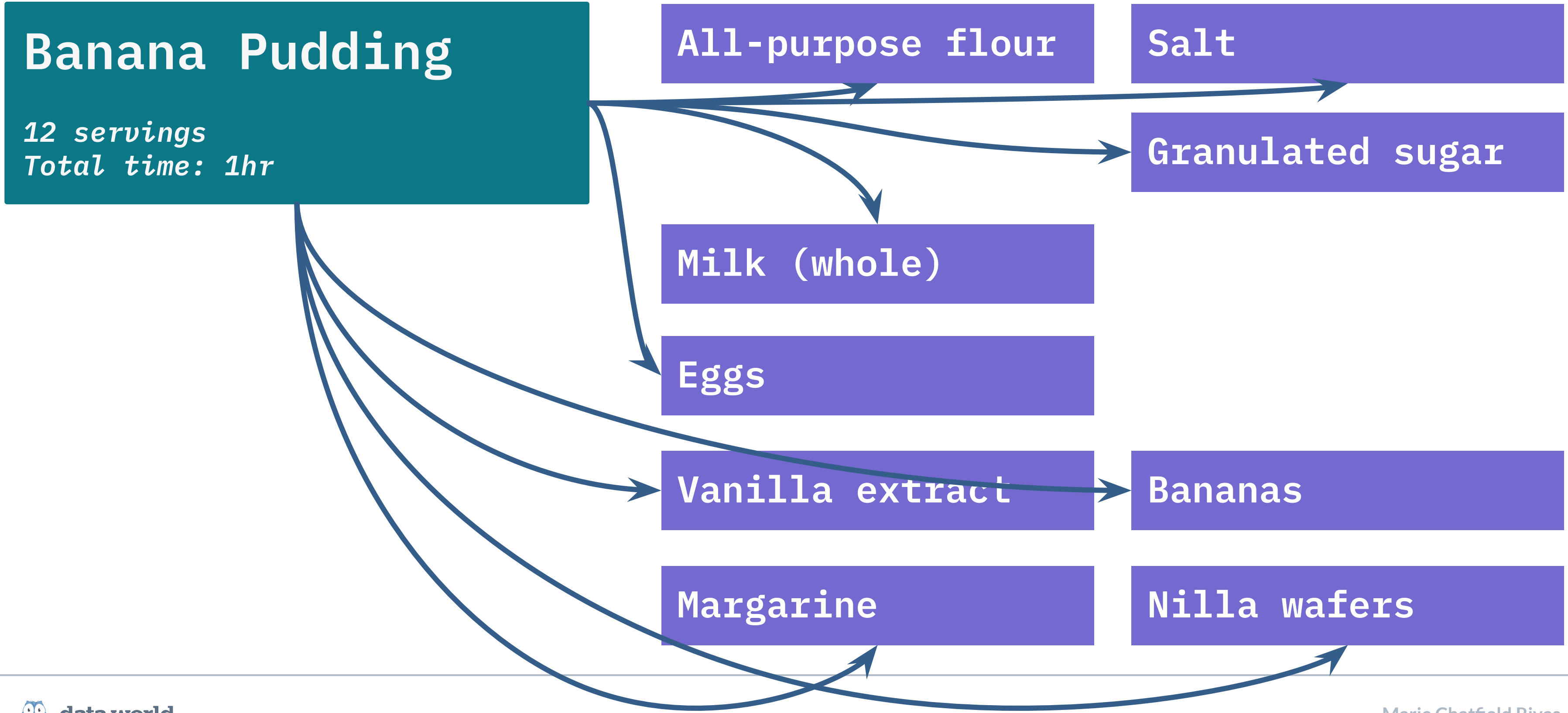
Vanilla extract

Bananas

Margarine

Nilla wafers

Update knowledge graph



Banana Pudding

12 servings
Total time: 1hr

All-purpose flour

Salt

Granulated sugar

Milk (whole)

Eggs

Vanilla extract

Bananas

Margarine

Nilla wafers

Parsers convert **structured text** into **well-defined values.**

Should I parse?

If you expect the LLM to provide **structured data** rather than raw text.

Post-process for reliability

Ingredients

- All-purpose flour
- Bread flour
- Milk (whole)
- Eggs
- Vanilla extract
- Margarine
- Salt
- Granulated sugar
- Brown sugar
- Chocolate chips
- Bananas
- Nilla wafers

Ingredients

- Refined sugar
- Table sugar
- White sugar
- Sugar

- All-purpose flour
- Bread flour
- Milk (whole)
- Eggs
- Vanilla extract
- Margarine

- Salt
- **Granulated sugar**
- Brown sugar
- Chocolate chips
- Bananas
- Nilla wafers

Large Language Model

This is a recipe. Return standardized ingredients as valid JSON in the format: { "ingredients": [] }.

You know about the following standard ingredients. If an ingredient is not on this list, you may propose a new one, using the most generic name.

```
[
  "All-purpose flour",
  "Bread flour",
  "Brown sugar",
  "Granulated sugar",
  …
]
```

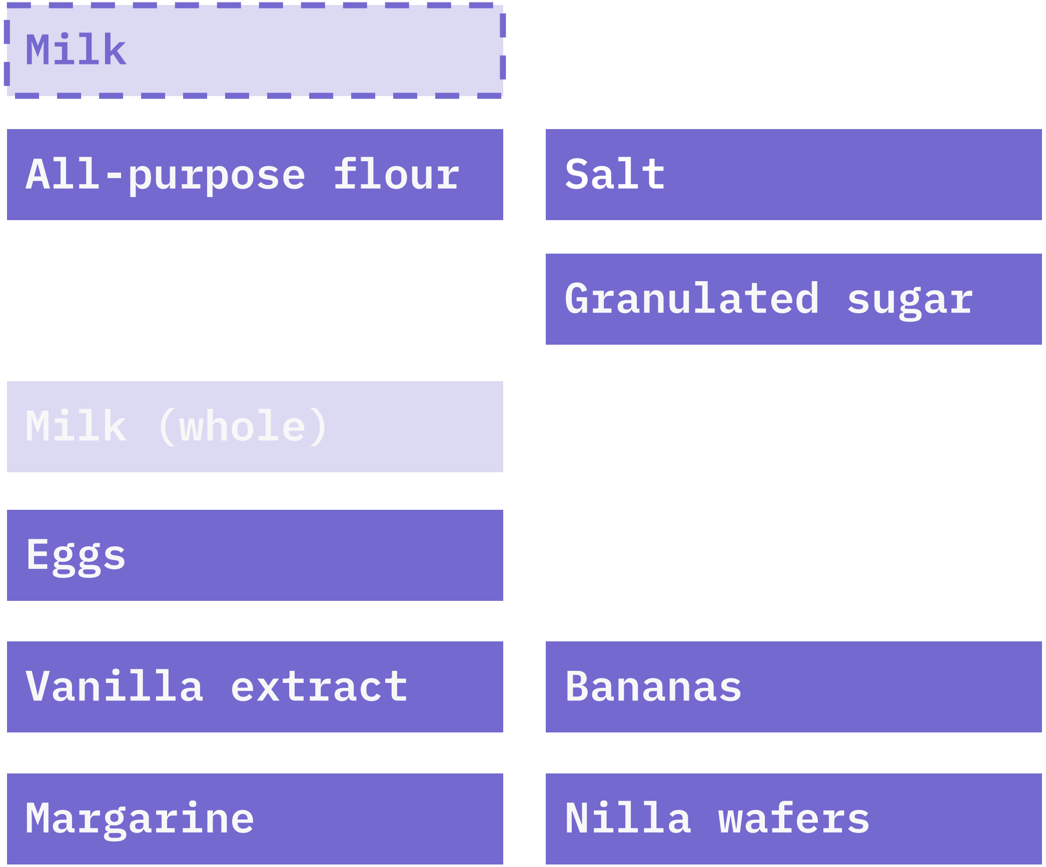
```
Banana Pudding

1/3 cup flour
2/3 cup sugar
1/4 teaspoon salt
2 eggs
…
```

```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk (whole)",
    "Margarine",
    "Vanilla extract",
    "Bananas",
    "Nilla wafers"
  ]
}
```

All-purpose flour
Salt
Granulated sugar
Milk (whole)
Eggs
Vanilla extract
Bananas
Margarine
Nilla wafers

```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk",
    "Margarine",
    "Vanilla extract",
    "Bananas",
    "Nilla wafers"
  ]
}
```



Milk

All-purpose flour

Salt

Granulated sugar

Milk (whole)

Eggs

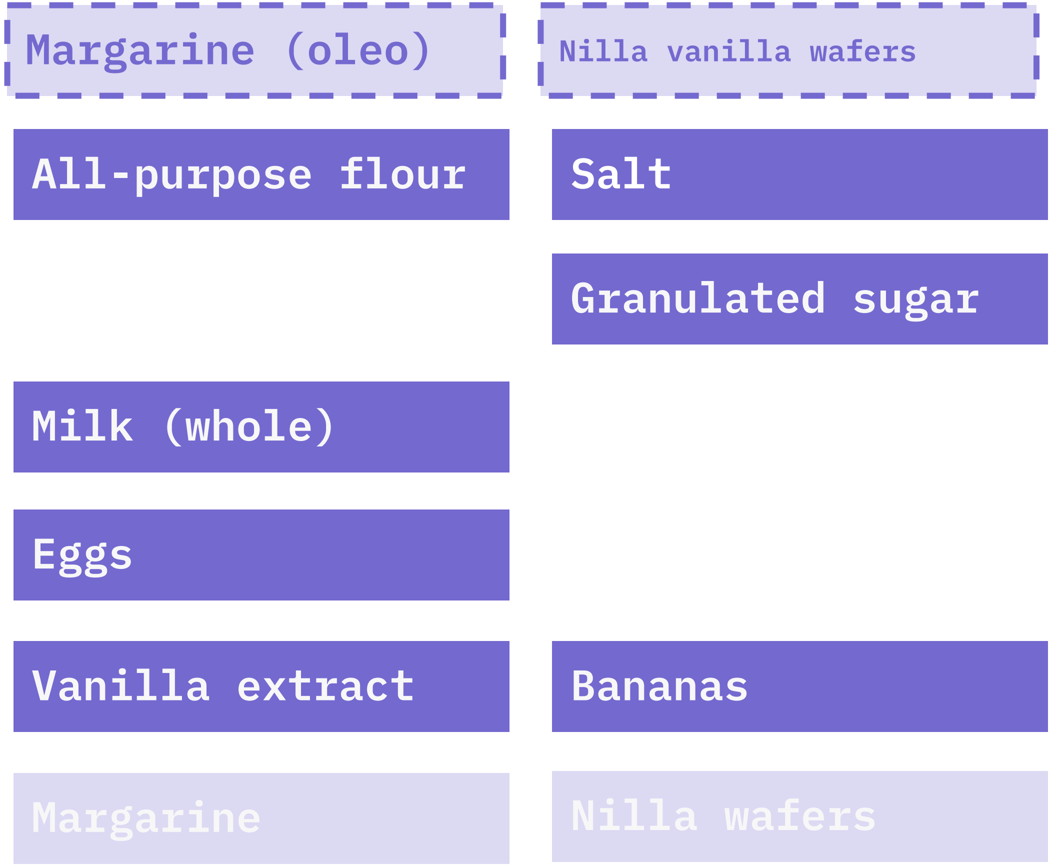
Vanilla extract

Bananas

Margarine

Nilla wafers

```
{
  "ingredients": [
    "All-purpose flour",
    "Granulated sugar",
    "Salt",
    "Eggs",
    "Milk (whole)",
    "Margarine (oleo)",
    "Vanilla extract",
    "Bananas",
    "Nilla vanilla wafers"
  ]
}
```



Margarine (oleo)

Nilla vanilla wafers

All-purpose flour

Salt

Granulated sugar

Milk (whole)

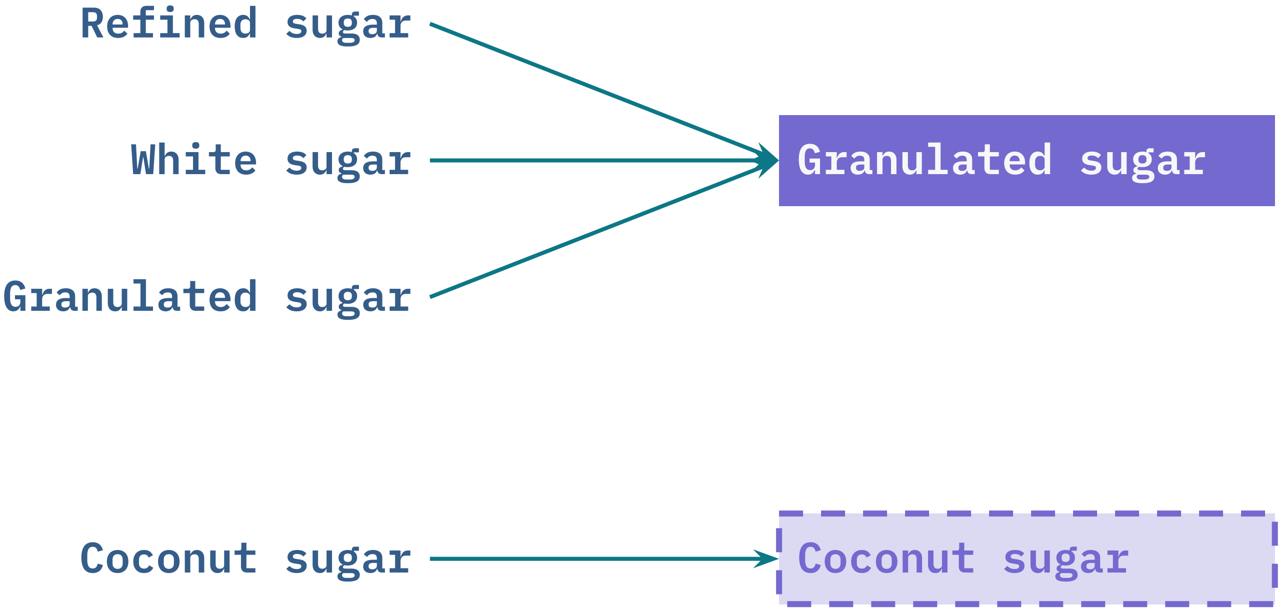
Eggs

Vanilla extract

Bananas

Margarine

Nilla wafers



Refined sugar

White sugar

Granulated sugar

Granulated sugar

Coconut sugar

Coconut sugar

Post-processing reliably increases the “**good enough**” target size.

Should I prompt or post-process?

Prompt for main use case.

Post-process for **edge cases and known issues.**

Craft domain-specific grammars

Chocolate Pie

8 servings

Meringue (below Filling)
Filling
1 cup sugar
3 Tablespoons Argo cornstarch
2 level Tablespoons Hershey's cocoa
1 regular size can Pet evaporated milk (12 oz.)
1 cup water
3 egg yolks
1 Tablespoon vanilla
2 Tablespoons oleo or margarine (firm vegetable oil product)

Meringue
3 egg whites (left from when you separated the eggs for the pie filling)
6 Tablespoons white sugar
1/4 teaspoon cream of tartar

Chocolate Pie

8 servings

Meringue (below Filling)
Filling
1 cup sugar
3 Tablespoons **Argo** cornstarch
2 level Tablespoons **Hershey’s** cocoa
1 regular size can **Pet** evaporated milk (12 oz.)
1 cup water
3 egg yolks
1 Tablespoon vanilla
2 Tablespoons oleo or margarine (firm vegetable oil product)

Meringue
3 egg whites (left from when you separated the eggs for the pie filling)
6 Tablespoons white sugar
1/4 teaspoon cream of tartar

"Evaporated milk"

```
{
  "name": "Evaporated milk",

  "substitutions": [
    "Milk (whole)"
  ],

  "brands": ["Pet"],

  "create": true
}
```

Large Language Model

This is a recipe. Return standardized ingredients without measurements as valid JSON in the format:

```
{
  "ingredients": [
     { "name": "", "create": false, "substitutions": [], "brands": [] }
  ]
}
```

```
{"ingredients":[
  {"name":"Granulated sugar","create":false,"substitutions":[],"brands":[]},
  {"name":"Cornstarch","create":true,"substitutions":[],"brands":["Argo"]},
  {"name":"Cocoa","create":true,"substitutions":[],"brands":["Hershey’s"]},
  {"name":"Evaporated milk","create":false,"substitutions":["Milk (whole)"],"brands":["Pet"]},
  {"name":"Egg yolks","create":false,"substitutions":[],"brands":[]},
  {"name":"Vanilla extract","create":false,"substitutions":[],"brands":[]},
  {"name":"Margarine","create":false,"substitutions":["Butter"],"brands":[]},
  {"name":"Egg whites","create":true,"substitutions":[],"brands":[]},
  {"name":"Cream of tartar","create":true,"substitutions":[],"brands":[]}
]}
```

```
{ "ingredients":
  [
    { "name":"Granulated sugar","create":false,"substitutions":[],"brands":[] },
    { "name":"Cornstarch","create":true,"substitutions":[],"brands":["Argo"] },
    { "name":"Cocoa","create":true,"substitutions":[],"brands":["Hershey’s"] },
    {
      "name":"Evaporated milk",
      "create":false,
      "substitutions":["Milk (whole)"],
      "brands":["Pet"]
    },
    { "name":"Egg yolks","create":false,"substitutions":[],"brands":[] },
    { "name":"Vanilla extract","create":false,"substitutions":[],"brands":[] },
    { "name":"Margarine","create":false,"substitutions":["Butter"],"brands":[] },
    { "name":"Egg whites","create":true,"substitutions":[],"brands":[] },
    { "name":"Cream of tartar","create":true,"substitutions":[],"brands":[] }
  ]
}
```

```
Granulated sugar
! Cornstarch | [Argo]
! Cocoa | [Hershey’s]
Evaporated milk | <Milk (whole)> [Pet]
Egg yolks
Vanilla extract
Margarine | <Butter>
! Egg whites
! Cream of tartar
```

Custom Grammar

```
Granulated sugar
! Cornstarch | [Argo]
! Cocoa | [Hershey’s]
Evaporated milk | <Milk (whole)> [Pet]
Egg yolks
Vanilla extract
Margarine | <Butter>
! Egg whites
! Cream of tartar
```

```
<ingredients> ::=
    <ingredient> (<newline> <ingredient>)*

<ingredient>  ::=
    <create>? <phrase> ("|" <config>)?

<create>       ::= "!"

<config>       ::= <substitution>* <brand>*
<substitution> ::= "<" <phrase> ">"
<brand>        ::= "[" <phrase> "]"
```

Large Language Model

This is a recipe. Return standardized ingredients without measurements, using this ingredient grammar:

```
<ingredients> ::= <ingredient> <newline>? <ingredients>?
<ingredient> ::= <create>? <phrase> ("|" <config>)?
<create> ::= "!"
<config> ::= (<space> <substitution>)* (<space> <brand>)*
<substitution> ::= "<" <phrase> ">"
<brand> ::= "[" <phrase> "]"
```

Large Language Model

This is a recipe. Return standardized ingredients without measurements, using this ingredient grammar:

```
- Each Ingredient is specified by a single line of text which begins with its Name.
- Names should be simple and generic, without any brand names.
- An Ingredient that is not yet defined MUST prefix its name with an exclamation point and space: !
- Ingredient MAY have a pipe separator: | followed by Substitutions then Brands.
- Substitutions are similar ingredients that may be used instead, and are wrapped in angle brackets: < >
- Brands are the specific products the recipe suggests, and are wrapped in square brackets: [ ]
```

Large Language Model

This is a recipe. Return standardized ingredients without measurements, using this ingredient grammar:

- Each Ingredient is specified by a single line of text which begins with its Name.
- Names should be simple and generic, without any brand names.
- An Ingredient that is not yet defined MUST prefix its name with an exclamation point and space: !
- Ingredient MAY have a pipe separator: | followed by Substitutions then Brands.
- Substitutions are similar ingredients that may be used instead, and are wrapped in angle brackets: < >
- Brands are the specific products the recipe suggests, and are wrapped in square brackets: [ ]

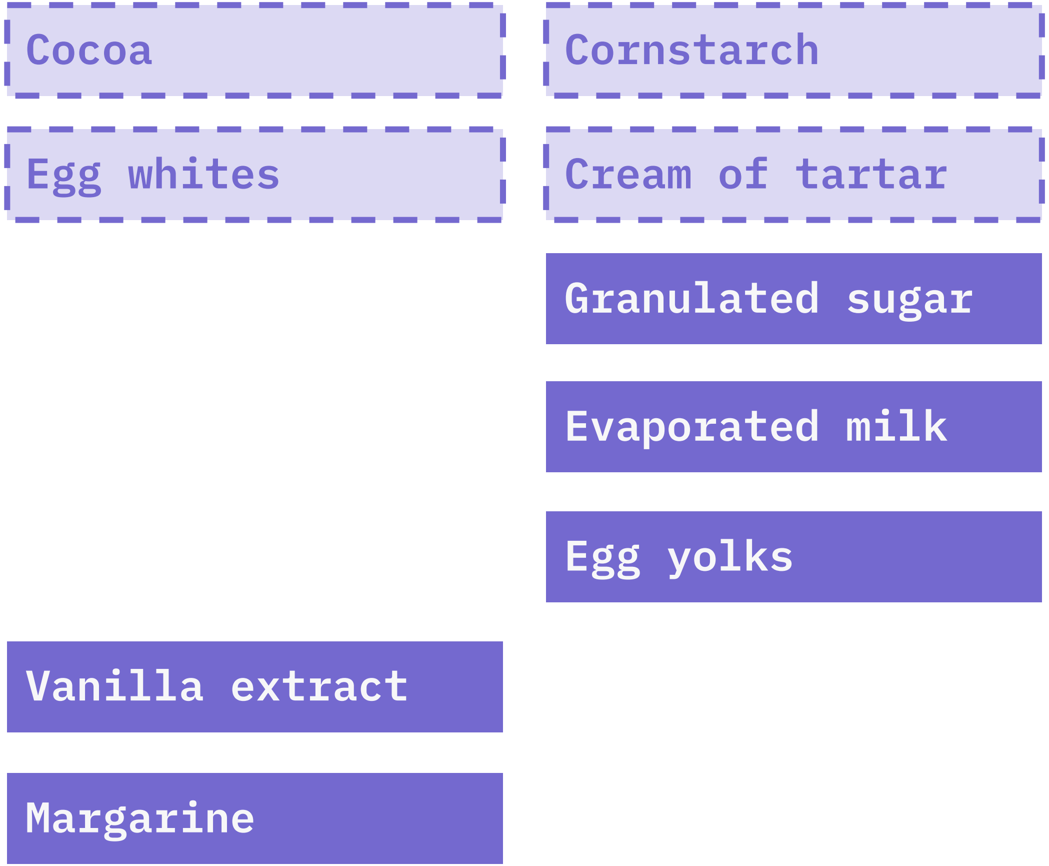
These ingredients already exist:

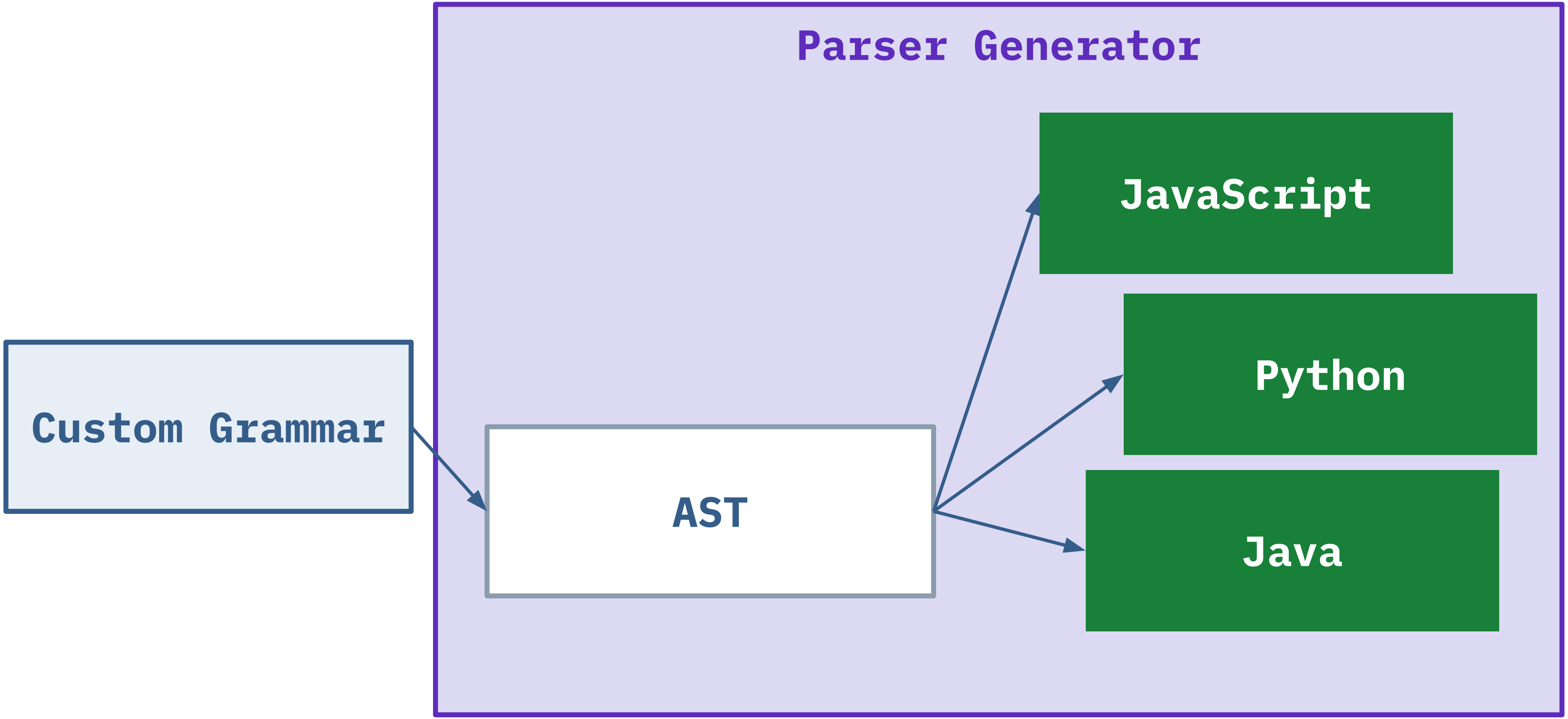
```
All-purpose flour | <Whole-wheat flour> [King Arthur Baking]
Bread flour | [King Arthur Baking]
Brown sugar
Granulated sugar
Milk (whole)
…
```

customParser.parse(

```
Granulated sugar
! Cornstarch | [Argo]
! Cocoa | [Hershey's]
Evaporated milk | <Milk (whole)>
[Pet]
Egg yolks
Vanilla extract
Margarine | <Butter>
! Egg whites
! Cream of tartar
```

)





Parser Generator

Custom Grammar

AST

JavaScript

Python

Java

Generate ingredients with custom grammar

Chocolate Pie

8 servings

Meringue (below Filling)
Filling
1 cup sugar
3 Tablespoons Argo cornstarch
2 level Tablespoons Hershey’s cocoa
1 regular size can Pet evaporated milk (12 oz.)
1 cup water
3 egg yolks
1 Tablespoon vanilla
…

Large Language Model

```
Granulated sugar
! Cornstarch | [Argo]
! Cocoa | [Hershey’s]
Evaporated milk | <Milk (whole)> [Pet]
Egg yolks
Vanilla extract
Margarine | <Butter>
! Egg whites
! Cream of tartar
```

Parse custom grammar

Chocolate Pie

8 servings

Cocoa

Cornstarch

Egg whites

Cream of tartar

Butter

Sugar

Milk (whole)

Evaporated milk

Egg yolks

Hershey’s

Argo

Pet

Vanilla extract

Margarine

Post-process & validate

Chocolate Pie

8 servings

Cocoa

Cornstarch

Egg whites

Cream of tartar

Butter

Granulated sugar

Milk (whole)

Evaporated milk

Egg yolks

Hershey's

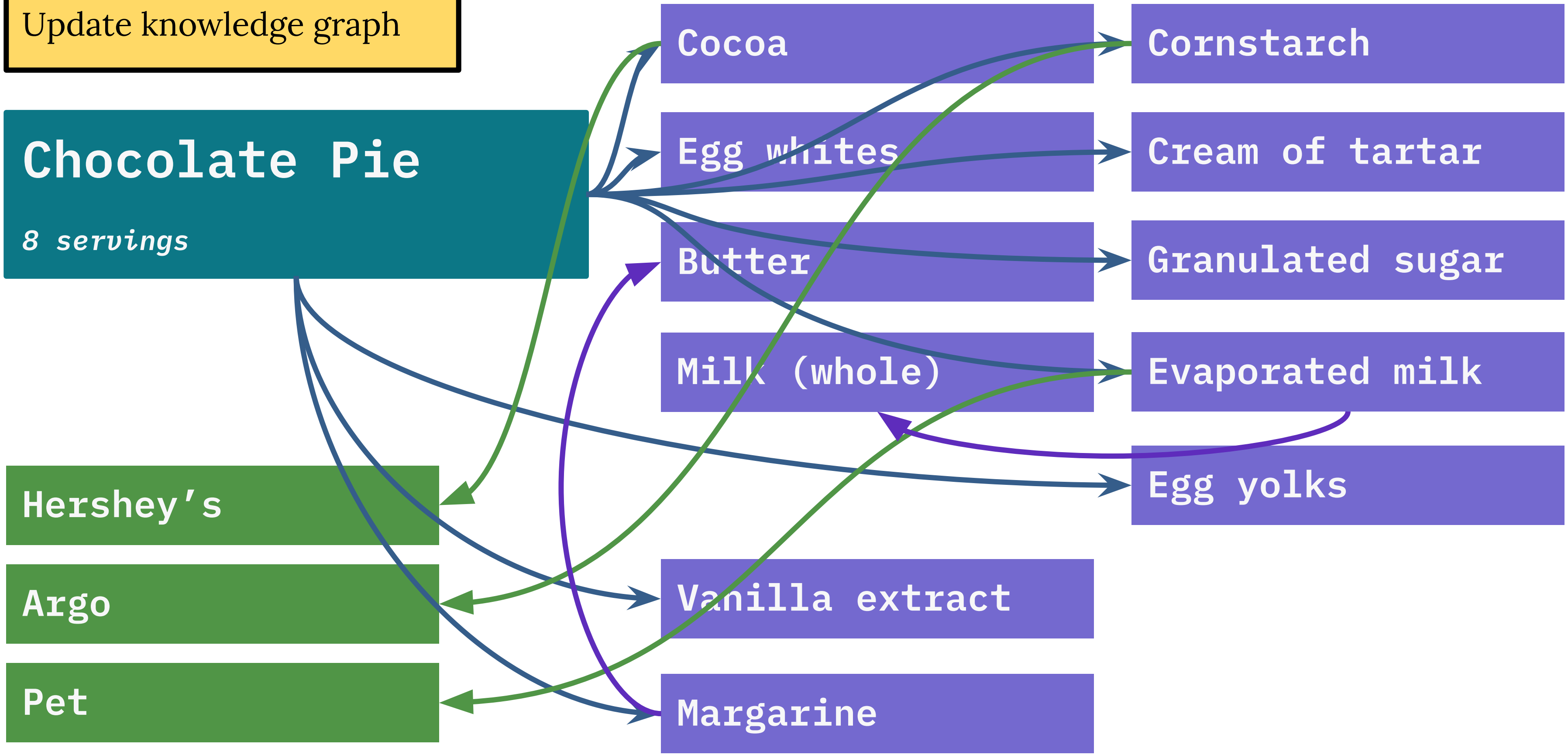
Argo

Pet

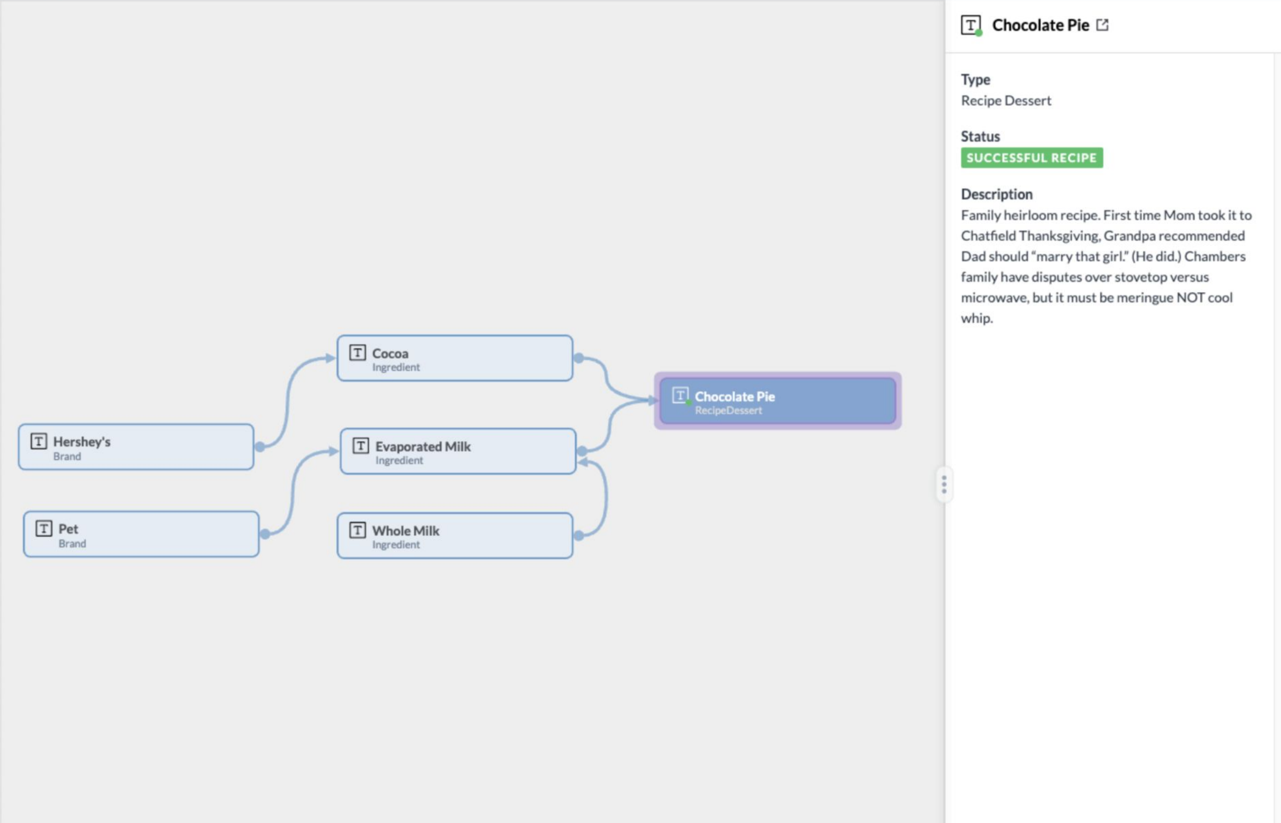
Vanilla extract

Margarine

Update knowledge graph



data.world



Chocolate Pie

Type
Recipe Dessert

Status
SUCCESSFUL RECIPE

Description
Family heirloom recipe. First time Mom took it to Chatfield Thanksgiving, Grandpa recommended Dad should "marry that girl." (He did.) Chambers family have disputes over stovetop versus microwave, but it must be meringue NOT cool whip.

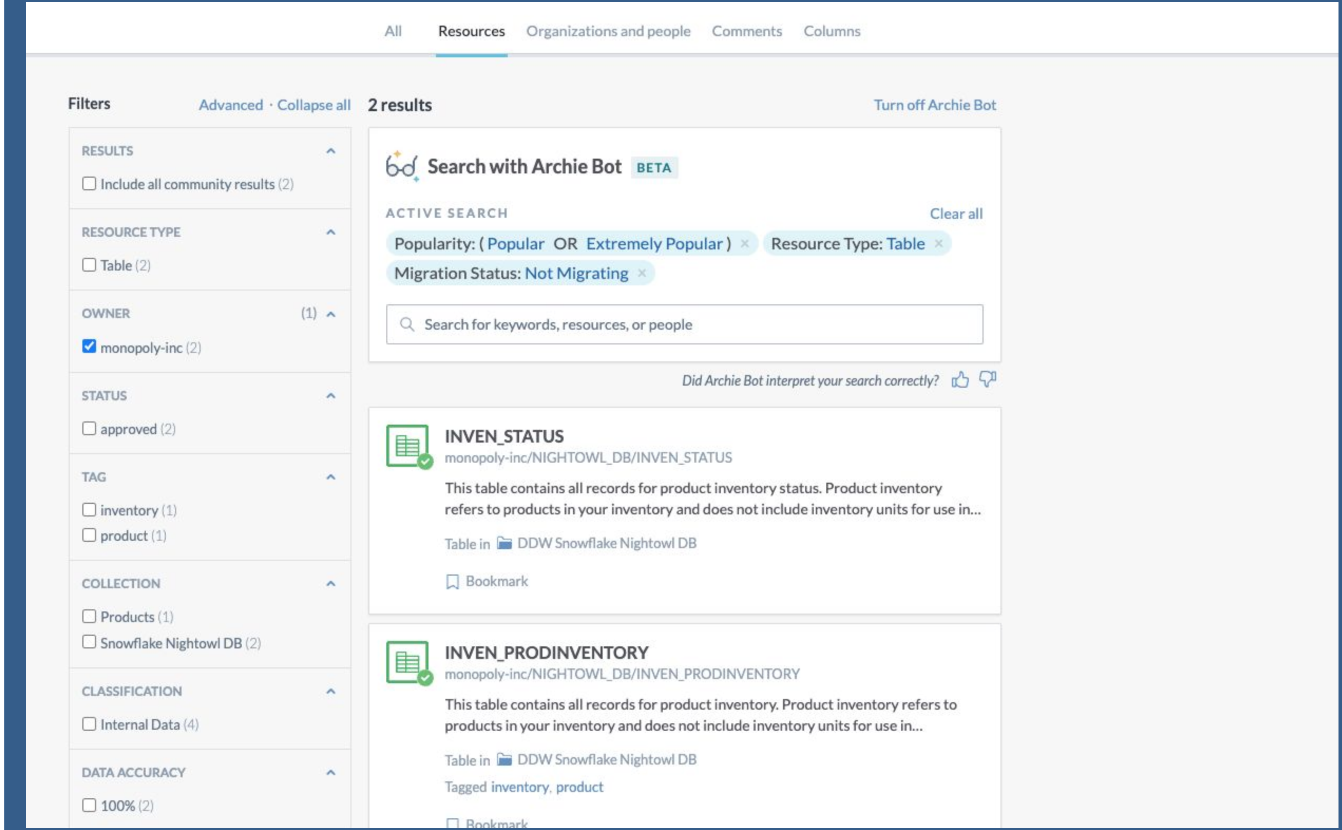
Domain-specific grammars can be more **precise** ***and*** **concise.**

Should I write a custom grammar?

If your domain is **difficult to model** with standard formats or requires **precise** and **complex syntax**.

Optimize grammars for LLM cooperation

data.world



All · Resources · Organizations and people · Comments · Columns

Filters Advanced · Collapse all

2 results Turn off Archie Bot

RESULTS
- ☐ Include all community results (2)

RESOURCE TYPE
- ☐ Table (2)

OWNER (1)
- ☑ monopoly-inc (2)

STATUS
- ☐ approved (2)

TAG
- ☐ inventory (1)
- ☐ product (1)

COLLECTION
- ☐ Products (1)
- ☐ Snowflake Nightowl DB (2)

CLASSIFICATION
- ☐ Internal Data (4)

DATA ACCURACY
- ☐ 100% (2)

Search with Archie Bot BETA

ACTIVE SEARCH · Clear all

Popularity: (Popular OR Extremely Popular) × Resource Type: Table × Migration Status: Not Migrating ×

Search for keywords, resources, or people

Did Archie Bot interpret your search correctly?

INVEN_STATUS
monopoly-inc/NIGHTOWL_DB/INVEN_STATUS
This table contains all records for product inventory status. Product inventory refers to products in your inventory and does not include inventory units for use in...
Table in DDW Snowflake Nightowl DB
Bookmark

INVEN_PRODINVENTORY
monopoly-inc/NIGHTOWL_DB/INVEN_PRODINVENTORY
This table contains all records for product inventory. Product inventory refers to products in your inventory and does not include inventory units for use in...
Table in DDW Snowflake Nightowl DB
Tagged inventory, product
Bookmark

status Approved Deprecated Pending

meta_Popularity Unpopular Popular Extremely Popular

meta_Sensitivity PII PCI PIFI

has status meta_Popularity meta_Sensitivity `status` `Approved` `Deprecated` `Pending`

`metadata`

- `Popularity` `Unpopular` `Popular` `Extremely Popular`
- `Sensitivity` `PII` `PCI` `PIFI`

`has` `status` `meta_Popularity` `meta_Sensitivity` `status` `Approved` `Deprecated` `Pending`

`metadata`

- `Popularity` `Unpopular` `Popular` `Extremely Popular`
- `Sensitivity` `PII` `PCI` `PIFI`

`has` `status`

`hasMetadata` `Popularity` `Sensitivity` status Approved Deprecated Pending

metadata

Popularity https://monopoly-inc.data.world/Unpopular https://monopoly-inc.data.world/Popular https://monopoly-inc.data.world/Extremely_Popular

Sensitivity PII PCI PIFI

has status

hasMetadata Popularity Sensitivity

status
Approved
Deprecated
Pending

resourcetype
businessTerm
dataset
catalogEntry

type
Metric
Table
Tableau Dashboard

metadata

Popularity
https://monopoly-inc.data.world/Unpopular
https://monopoly-inc.data.world/Popular
https://monopoly-inc.data.world/Extremely_Popular

Sensitivity
PII
PCI
PIFI

has
status

hasMetadata
Popularity
Sensitivity

Large Language Model

You are a filtering assistant and you return DW-SEARCH.

DW-SEARCH uses AND, NOT, OR. There are NO asterisks. These are the filters:

status Approved Deprecated Pending

resourcetype businessTerm dataset catalogEntry

type Metric Table Tableau Dashboard

metadata

Popularity https://monopoly-inc.data.world/Unpopular https://monopoly-inc.data.world/Popular https://monopoly-inc.data.world/Extremely_Popular

Sensitivity PII PCI PIFI

has status

hasMetadata Popularity Sensitivity

Large Language Model

You are a filtering assistant and you return DW-SEARCH.

DW-SEARCH uses AND, NOT, OR. There are NO asterisks. These are the filters:

status	Approved	Deprecated	Pending
meta_Popularity	Unpopular	Popular	Extremely Popular
meta_Sensitivity	PII	PCI	PIFI
has	status	meta_Popularity	meta_Sensitivity

Generate intermediary grammar

Popular tables without any kind of status

Large Language Model

```
type:"table"

AND meta_Popularity:"Popular"

AND NOT (has:"meta_MigrationStatus"
         OR has:"status")
```

Parse

Popular tables without any kind of status

`type:"table"`

AND

`meta_Popularity:"Popular"`

AND NOT

- `has:"meta_MigrationStatus"`

  OR

- `has:"status"`

Translate to standard grammar

Popular tables without any kind of status

`type:"table"`

AND

`metadata:"Popularity:https://monopoly-inc.data.world/Popular"`

AND NOT

`hasMetadata:"Popularity"`

OR

`has:"status"`

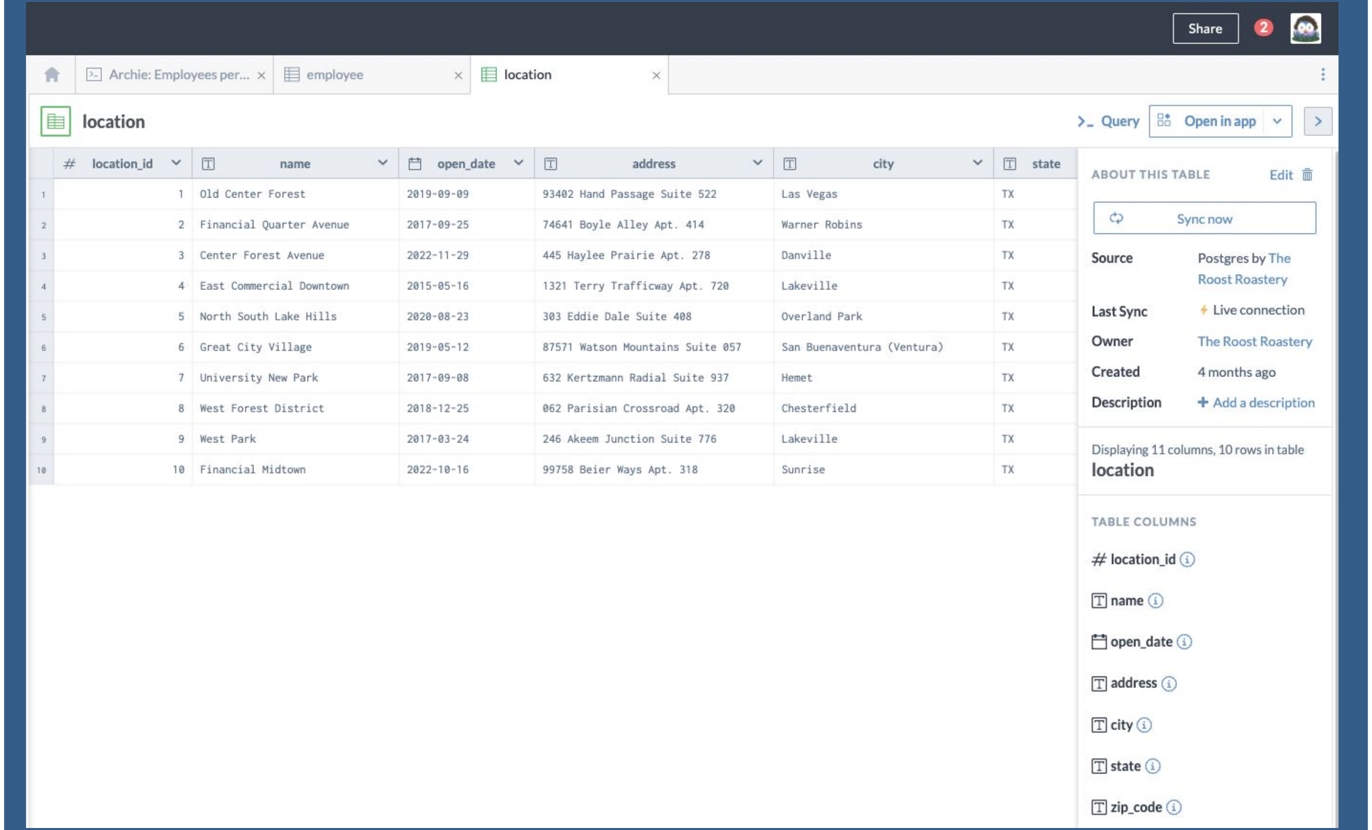
Work with the LLM by engineering a grammar according to its **response patterns.**

Should I fine-tune a model on my grammar?

Only if the standard model fails when prompted with **a subset** or **simpler grammar**.

Support multiple languages with post-processing

data.world



location

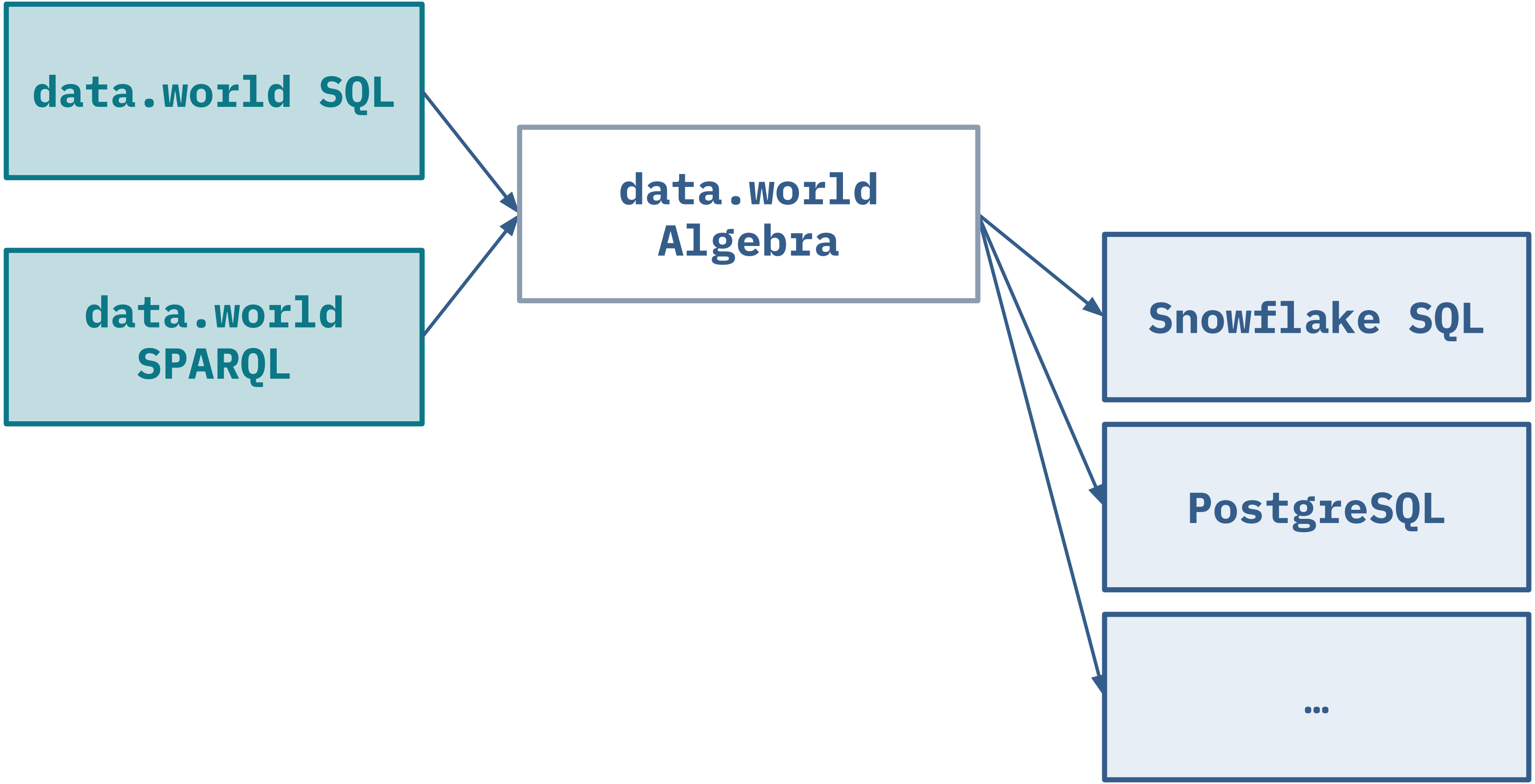
	location_id	name	open_date	address	city	state
1	1	Old Center Forest	2019-09-09	93402 Hand Passage Suite 522	Las Vegas	TX
2	2	Financial Quarter Avenue	2017-09-25	74641 Boyle Alley Apt. 414	Warner Robins	TX
3	3	Center Forest Avenue	2022-11-29	445 Haylee Prairie Apt. 278	Danville	TX
4	4	East Commercial Downtown	2015-05-16	1321 Terry Trafficway Apt. 720	Lakeville	TX
5	5	North South Lake Hills	2020-08-23	303 Eddie Dale Suite 408	Overland Park	TX
6	6	Great City Village	2019-05-12	87571 Watson Mountains Suite 057	San Buenaventura (Ventura)	TX
7	7	University New Park	2017-09-08	632 Kertzmann Radial Suite 937	Hemet	TX
8	8	West Forest District	2018-12-25	062 Parisian Crossroad Apt. 320	Chesterfield	TX
9	9	West Park	2017-03-24	246 Akeem Junction Suite 776	Lakeville	TX
10	10	Financial Midtown	2022-10-16	99758 Beier Ways Apt. 318	Sunrise	TX

Source	Postgres by The Roost Roastery
Last Sync	Live connection
Owner	The Roost Roastery
Created	4 months ago
Description	+ Add a description

Displaying 11 columns, 10 rows in table

location

- location_id
- name
- open_date
- address
- city
- state
- zip_code



data.world SQL

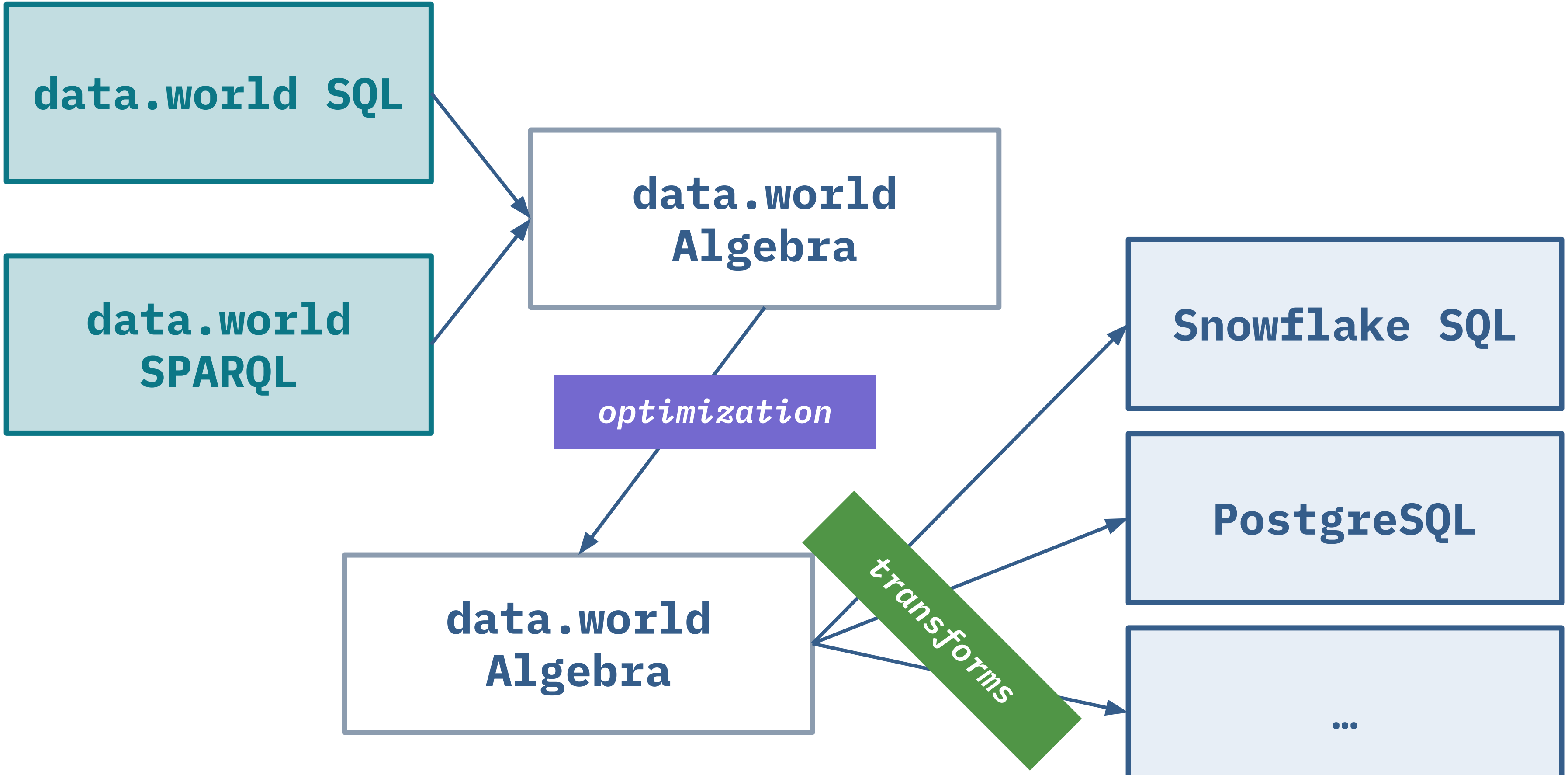
data.world SPARQL

data.world Algebra

Snowflake SQL

PostgreSQL

...



data.world SQL

data.world SPARQL

data.world Algebra

optimization

data.world Algebra

transforms

Snowflake SQL

PostgreSQL

…

algebra/optimization/

ExpressionRules.scala

```
// Rewrite constant value expressions to the equivalent constant.
//   SELECT * FROM Foo WHERE x > (2 + 2)
//     ↓
//   SELECT * FROM Foo WHERE x > 4
```

JoinRules.scala

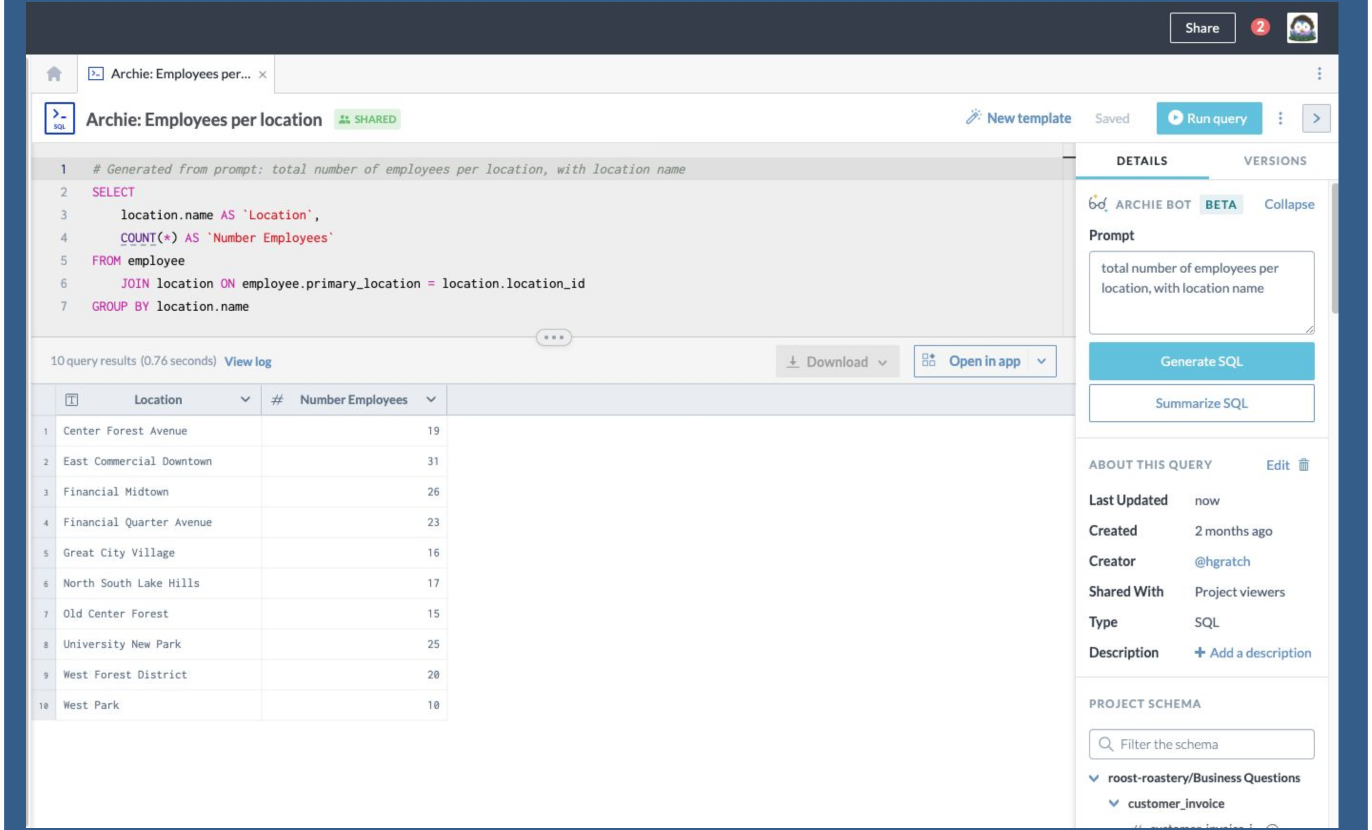
```
// Rewrite joins that are filtered into inner joins on those values.
//   SELECT * from Foo, Bar where Foo.baz = Bar.barangus
//     ↓
//   SELECT * from Foo join Bar on Foo.baz = Bar.barangus
```

sql/transforms/

DateSubToDateAddFixer.scala

```
// If a datasource implements DATE_ADD but not DATE_SUB,
// rewrite DATE_SUB in terms of negative DATE_ADD.
//
//   DATE_SUB(playerId, x, "days")
//     ↓
//   DATE_ADD(playerId, -x, "days")
```

data.world



Archie: Employees per location

SHARED

```
# Generated from prompt: total number of employees per location, with location name
SELECT
    location.name AS `Location`,
    COUNT(*) AS `Number Employees`
FROM employee
    JOIN location ON employee.primary_location = location.location_id
GROUP BY location.name
```

10 query results (0.76 seconds) View log

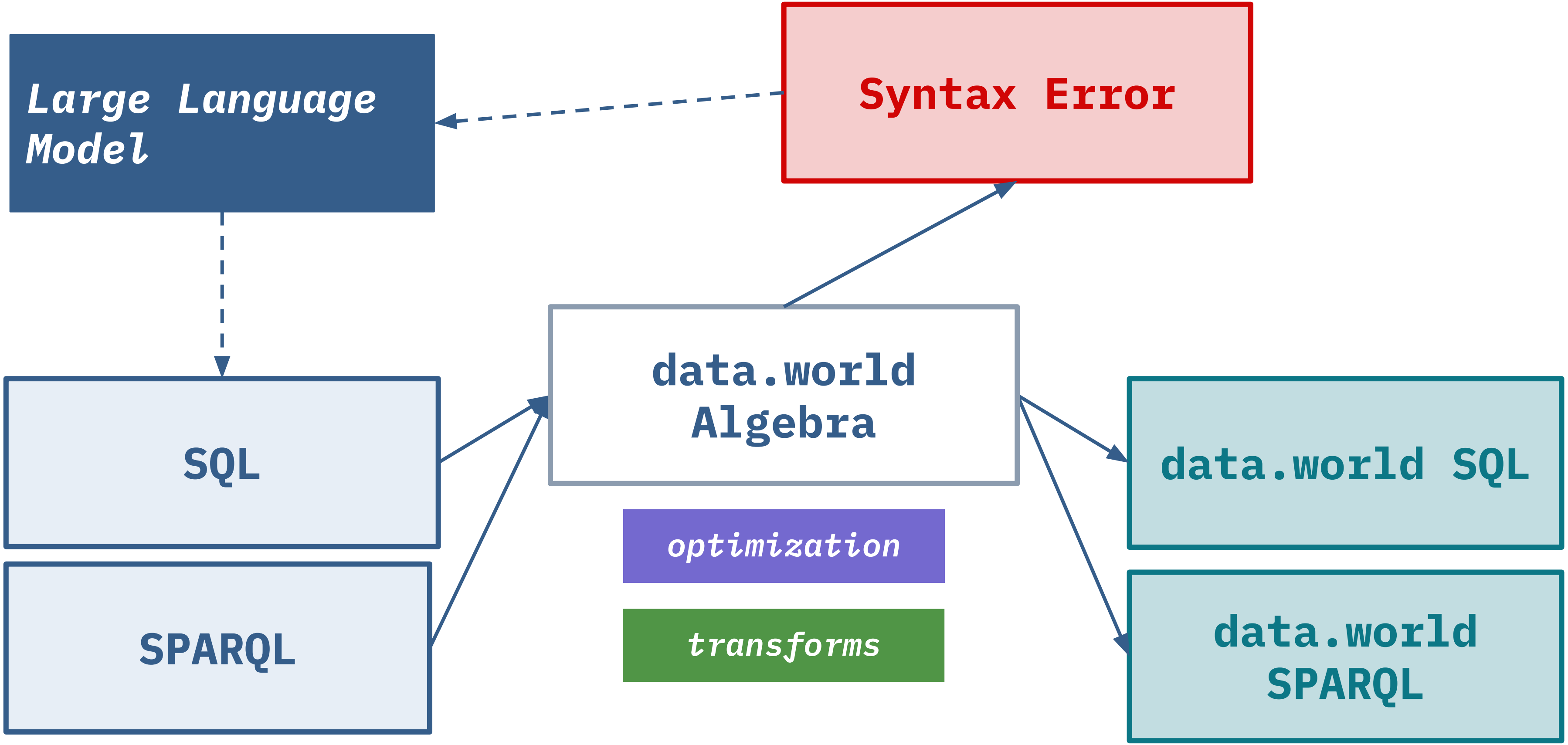
	Location	Number Employees
1	Center Forest Avenue	19
2	East Commercial Downtown	31
3	Financial Midtown	26
4	Financial Quarter Avenue	23
5	Great City Village	16
6	North South Lake Hills	17
7	Old Center Forest	15
8	University New Park	25
9	West Forest District	20
10	West Park	10

Prompt

total number of employees per location, with location name

ABOUT THIS QUERY

Last Updated	now
Created	2 months ago
Creator	@hgratch
Shared With	Project viewers
Type	SQL
Description	+ Add a description


Large Language Model
Syntax Error
SQL
SPARQL
data.world Algebra
optimization
transforms
data.world SQL
data.world SPARQL

sql/transforms/

IncorrectQualifierReconciler.scala

```
// If the LLM hallucinates schema qualifiers, rewrite them against
// the known schema and context.
//
//   SELECT * FROM bad.worse.Bar
//     ↓
//   SELECT * FROM Bar
```

Generate SQL

`total number of employees per location, with location name`

Large Language Model

```sql
WITH employee_location AS (
  SELECT
    e.location_id,
    l.name AS location_name,
    COUNT(e.employee_id) AS total_employees
  FROM
    employee e
    JOIN location l ON e.location_id = l.location_id
  GROUP BY
    e.location_id,
    l.name
)
SELECT
  location_id,
  location_name,
  total_employees
FROM
  employee_location
```

Identify error in query

total number of employees per location, with location name

Large Language Model

```
WITH employee_location AS (
  SELECT
    e.location_id,
    l.name AS location_name,
    COUNT(e.employee_id) AS total_employees
  FROM
    employee e
    JOIN location l ON e.location_id = l
  GROUP BY
    e.location_id,
    l.name
)
SELECT
  location_i
  location_na
  total_employ
FROM
  employee_location
```

employee.location_id does not exist

Re-generate SQL

total number of employees per location, with location name

Large Language Model

```
WITH employee_count AS (
  SELECT l.location_id, l.name AS location_name,
      COUNT(*) AS total_employees
  FROM location l
  JOIN employee e ON l.location_id = e.primary_location
  GROUP BY l.location_id, l.name
)
SELECT location_id, location_name, total_employees
FROM employee_count
```

Parse and format query

total number of employees per location, with location name

Large Language Model

```
WITH employee_count AS (
    SELECT
        l.location_id AS `Location Id`,
        l.name AS `Location Name`,
        COUNT(*) AS `Total Employees`
    FROM location AS l
        JOIN employee AS e
          ON l.location_id = e.primary_location
    GROUP BY l.location_id, l.name
)

SELECT
  `Location Id`,
  `Location Name`,
  `Total Employees`
 FROM employee_count
```

Parsing into a **higher-order AST** enables multi-lingual translations.

Should I target a higher-order AST?

If you need to support multiple languages that **share meanings** but use **different syntax**.

Parse **standard grammars**

Post-process for **reliability**

Craft **domain-specific** grammars

Optimize grammars for **LLM cooperation**

Support **multiple languages** with post-processing

Learn less. Parse more.